AD-A073 346 NAVAL RESEARCH LAB WASHINGTON DC F/G 22/3

A NUMERICAL SOLUTION FOR ROCKET ASCENT TRAJECTORY. (U)

AUG 79 M H REILLY

UNCLASSIFIED NRL-MR-4044 SBIE-AD-E000 314 NL

1 OF 1

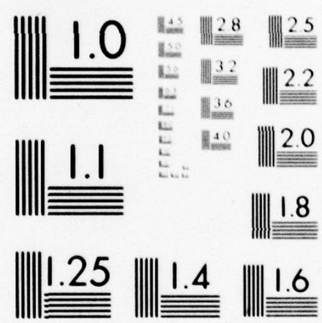
AD A073346

END

DATE FILMED

10-79

DDC



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU OF STANDARDS 1963-A

NRL Memorandum Report 4044

A Numerical Solution for Rocket Ascent Trajectory

MICHAEL H. REILLY

Telecommunications Environmental Effects Branch
Communications Sciences Division

August 16, 1979

NAVAL RESEARCH LABORATORY
Washington, D.C.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NRL Memorandum Report 4044	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A NUMERICAL SOLUTION FOR ROCKET ASCENT TRAJECTORY		5. TYPE OF REPORT & PERIOD COVERED Final report on one phase on a continuing NRL problem. 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Michael H. Reilly		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Research Laboratory Washington, DC 20375		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NRL Problem R04-16
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE August 16, 1979 13. NUMBER OF PAGES 65
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Rocket ascent program
Numerical solution for rocket trajectory

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A program for the numerical solution of rocket ascent trajectory is presented, which is particularly appropriate for large, multistage, unfinned rocket systems. Fortran listings of this program for DEC-10 and PDP-11/70 computer systems are given along with background information, examples, and prescriptions which clarify the usage of the program. Thrust increase and aerodynamic effects are specifically included, and the program is found to be both suitable and efficient for a wide variety of rocket trajectory problems. It is demonstrated how to modify the program results to include earth's rotation effects.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

CONTENTS

INTRODUCTION

In a previous report [1] the author presented a theoretical framework for analyzing the trajectory of rocket-satellite systems, both in the powered rocket ascent or launch phase and in the subsequent orbital motion or coast phase. Techniques for solving the problem of launching space systems into proper earth orbit were elucidated, and subsequently these techniques have been tried on particular cases. In the process, a numerical solution of the equations of motion of rocket ascent has been programmed on a digital computer and found to be useful. It is reported here to make it more generally available.

This is obviously not the first time a computer program has been developed to handle the rocket ascent problem. It is possible to get one of these programs from an outside source, and adapt it to the specific tasks at hand. Very often, however, the information accompanying the program is insufficient, or the requirements of the program are not matched to the data base for the problem or the computer at hand. It is frequently easier, faster, and more reliable to develop a program ab initio. This was done, and the resulting program is presented and explained below.

It was shown in Reference [1] that the rocket ascent problem could be solved as if the earth (and atmosphere) were stationary in an inertial frame of reference, and that the corrections from earth's rotation could be added in later. The computer program developed here solves the launch problem for the stationary earth case. It will be shown later in a specific application how earth's rotation effects are adjoined to the computer program results. For the stationary earth model the simplest case of planar rocket motion is shown in Fig. 1. Here $\underset{\sim}{r}$ is the radius vec-

Note: Manuscript submitted March 26, 1979.

tor from the center of the earth to the center of mass of the rocket, y is the rocket altitude, ϕ is the angular displacement from launch, which determines the ground range x when the radius of the earth R is factored in, ψ is the flight path angle of the center of mass (or "heading") of the rocket with respect to the local horizontal, and α is known as angle of attack of the rocket axis with respect to the flight direction.

The equations of motion for the rocket are deduced [1] straightforwardly from the force diagram shown in Fig. 2, which also contains a brief description of the indicated symbols. The forces indicated are: the gravitational force $\underset{\sim}{W}$, the thrust of the escaping exhaust gases $\underset{\sim}{T}$, which is canted at the angle β to the vehicle centerline, and the aerodynamic forces of drag $\underset{\sim}{D}$ and lift $\underset{\sim}{L}$ due to rocket motion through the atmosphere. Drag and lift forces are directed antiparallel and perpendicular to the flight path direction, respectively. In deducing the equations of motion, we make the assumptions found in many references [1-4]:

(1) the aerodynamic forces act through the center of pressure,

(2) the force of gravity acts through the center of gravity, and

(3) the thrust force is applied through the center of combustion.

The equations of motion for the center of mass motion are determined [1] to be:

$$\dot{v} = -(K/r^2)\sin\psi - (D/M) + (T/M)\cos(\alpha+\beta) \tag{1}$$

$$v\dot{\psi} = -\frac{K}{r^2} - \frac{v^2}{r}\cos\psi + \frac{T}{M}\sin(\alpha+\beta) + \frac{L}{M} \tag{2}$$

The speed of the rocket center of mass is here denoted by v, and the mass of the rocket is denoted by M. The symbol K is the product of the universal gravitational constant

and the mass of the earth, and has the value

$$K = 3.986012 \times 10^5 \ km.^3/sec.^2$$

The other symbols have already been defined. The angle of thrust β is normally very small for large launch vehicles (e.g., $\beta/\alpha \ll 1$), and it is customary [2,3] to set β to zero in the equations of motion before solving for the launch trajectory. Hence, the good approximation is made

$$\beta \approx 0 \tag{3}$$

in Eq.'s (1) and (2), thus avoiding the complications of additional moment equations for the steering of the rocket. The variation of β is preprogrammed in the guidance and control system of the rocket, which causes the relatively much larger variations in α involved in the shaping of the launch trajectory. The approach to be used in the numerical solution of Eq's (1) and (2) is in this spirit. An $\alpha(t)$ profile will be selected to cause the relatively much larger variations in $\psi(t)$, and it will be done in such a way as to obtain a correct value of ψ at rocket burnout and a smooth variation in $\psi(t)$. The program user typically knows this burnout value of ψ from consideration of the orbit into which he desires to inject the payload. The procedure will be: (1) selection of an $\alpha(t)$ profile, (2) numerical solution of $v(t)$ and $\psi(t)$ from the equations of motion, and then from orbit injection or other criteria, (3) selection of a new $\alpha(t)$ profile and a repetition of step (2). This process is iterated on the computer until a "correct" launch trajectory is obtained.

Other approximations in the solutions of Eq's (1) - (3) have been experimented with during the evolution of this program. One approximation is the choice of specific, analytic profile for $\psi(t)$ with free parameters adjusted so that $\psi(t)$ takes on specific end values. The known

end values are those at rocket launch and at orbit injection. The problem is that all the other values of $\psi(t)$ in between are forced by the analytic form; this will be elaborated on shortly. The advantage of assuming an analytic $\psi(t)$ profile is that only Eq. (1) needs to be solved if some other information about $\alpha(t)$ is known - e.g., that α is small enough so that $\cos \alpha \approx 1$ (recall Eq. (3)). The criterion of small α values is presumably desirable from the standpoint of launch efficiency; the rocket engine then does its work along the direction of motion. Ehricke [3] uses this approach, but points out that α could be calculated from $\psi(t)$ and the solution for v from Eq. (2). In particular, the assumption about the smallness of α could be checked. Ehricke uses a particular analytic form $\psi(t)$ for each stage of a multistage rocket. Experience with this approach, however, points out certain flaws in it. In the first place, the end points of $\psi(t)$, which nail down its free parameters, are typically unknown ahead of time. Hence, there is much time wasted experimenting with end values (i.e., stage ignition and burnout values of ψ) which will keep the range of α values small. A major problem is that the analytic form of $\psi(t)$ is apparently not very realistic, because it forces some rather wild variations of $\alpha(t)$. Neither is the criterion of smallness of $\alpha(t)$ necessarily realistic, particularly in the higher rocket stages when substantial tilts may be necessary to inject payloads at sufficiently high altitudes to avoid aerodynamic heating or drag effects. Ehricke's method was thus found to be too slow and inefficient in the trial and error aspect, not sufficiently unambiguous for inclusion of an automatic self-correcting procedure based on a small α criterion, and potentially too inaccurate for our purposes. Ruppe [3] similarly employs analytic forms for $\sin \psi$ and $\cos \psi$ which are polynomial expansions in t designed to give correct launch and orbit injection end point conditions. With

certain other approximations this enables one to integrate Eq. (1) analytically. Unfortunately, the large disagreements in the ψ values inferred from the separate polynomial expansions for sin ψ and cos ψ and the inaccuracies in the method seemed unacceptable for our purposes.

Other approximations, often made in the solution of Eq's (1) - (3), involve the evaluation of the force terms in these equations. The evaluation of forces in the program will be detailed in the next section, but for present purposes it will suffice to just mention the qualitative reasoning behind the approximations. Due to ambient pressure decrease with altitude it is found that the effective rocket thrust force T in Eq's (1) and (2) increases from its launch value T_{sl} at sea level to its vacuum value T_{vac} at the orbit injection altitude. This increase of thrust effect is at least partially offset by the drag force in Eq. (1) which acts to slow the rocket down. This has prompted certain authors [3,4] to eliminate the drag and thrust increase effects from Eq. (1), since they tend to counterbalance each other, and use a constant thrust term instead. This is a nice simplification, particularly since the drag and lift coefficients for the rocket, which enter in the evaluation of D and L in Eq's (1) and (2), are usually not known very well. Similarly, the force L is often dropped from Eq. (2) on the grounds that it is not very important for large rockets. It has been the author's experience that for large rockets, e.g., with liftoff weights of hundreds of thousands of lbs. or more, the thrust increase effect outweighs the drag effect. As a result, the constant thrust approximations give injection velocities smaller than the more complete evaluations by

as much as 2-3%. This kind of discrepancy is serious for the orbit determination, as shown in Fig. 3, which gives the dependence of orbit apogee altitude on perigee injection velocity. A two percent variation injection speed can make a difference of 500 nmi. in apogee altitude. In questions of orbit determination (or even whether or not a stable orbit has been achieved) this kind of uncertainty may well be unacceptable. As Eq. (1) indicates, the thrust term also varies when the angle of attack α becomes large, which is often the case in the upper rocket stages. The constant thrust approximation was therefore abandoned. The decision was also made to not discard the lift term L in Eq. (2), since it was found [3] to significantly enhance the leverage exerted by the assumed $\alpha(t)$ profile on the calculated $\psi(t)$ solution at ascent altitudes where aerodynamic effects attain their largest values.

For the above reasons the approach adopted in the computer program is to assume an $\alpha(t)$ profile, solve the equations of motion, and then refine the original $\alpha(t)$ in an iterative approach. This proved to be easy to automate, and the results did not appear to be sensitive to the ambiguities in choosing the $\alpha(t)$ profile. With an evaluation of thrust increase and aerodynamic effects, and with a non-reliance on the constant thrust approximation, the program has reached a level of approximation where it should be useful for most purposes, including orbit determination. More on this later, but the next task is to supply details about the program and its use.

The Evaluation of Terms in the Equations of Motion

For a rocket with swivel control motors the total thrust T in Eq's (1) and (2) includes pressure forces, and is written [3-5] as

$$T = \dot{M}v_e + A\,[p_e - p(y)] \quad , \qquad (4)$$

where $\dot{M}$ is the mass loss rate of exhaust gases, v_e is the actual average axial speed of these gases relative to the rocket, A is the exit aperture area, p_e is the pressure of the exhaust gases at the exit aperture, and p(y) is the ambient pressure of the atmosphere at altitude y. This pressure varies from p(o) to zero during the first part of rocket flight, usually by the time of first stage separation. Consequently, T varies from its sea level value T_{sl} to its vacuum value T_{vac}. Both T_{sl} and T_{vac} are normally specified for the first stage of the rocket, possibly also for the second stage, but usually only T_{vac} is given for higher stages. Hence, for the first stage, possibly also the second, the ratio

$$x \equiv T_{vac}/T_{sl} \qquad (5)$$

is known, and Eq. (4) can be rewritten as

$$T = T_{vac}\left[1 - \frac{x-1}{x}\;\frac{p(y)}{p\,(o)}\right] \qquad (6)$$

The computer program evaluates thrust from this equation, which exhibits the increase of thrust with altitude.

The aerodynamic forces of lift L and drag D in Eq's (1) and (2) can be evaluated from [3]:

$$D = [C_{Do} + C_{DL}\,\alpha^2]\;(\rho v^2/2)S \qquad (7)$$

$$L = [(\partial C_L/\partial\alpha)\;\alpha]\;(\rho v^2/2)S \qquad (8)$$

where ρ is the atmospheric mass density, v is the rocket speed and S is a reference cross-sectional area (e.g., that for the lowest rocket stage) to which the aerodynamic coefficients are referred. As seems plausible, the aerodynamic forces are seen to be proportional to the dynamic

pressure of the atmosphere (i.e., $\rho v^2/2$) and the reference cross-sectional area of the rocket vehicle. The proportionality constants in square brackets in Eq's (7) and (8) are the drag and lift coefficients which depend on both the angle of attack α and the speed v. The dependence on α is to be expected, since the tilting of the rocket centerline away from the flight direction obviously exposes more of the rocket surface to the pressure forces of the atmosphere, thus increasing the aerodynamic forces. The lift force vanishes for $\alpha = 0$. The representation of the α dependence of the drag and lift coefficients in Eq's (7) and (8) is supposed to be valid, according to Ehricke [3], for moderate angles of attack (e.g., $\alpha < 10^\circ$). Here C_{DO}, C_{DL}, and $\partial C_L/\partial \alpha$ are constants which depend only on the speed of the rocket. Actually, the dependence is found to be on the mach number M, defined as

$$M = v/c \,, \tag{9}$$

where c is the speed of sound at the altitude of the rocket. We have used the dependences given by Ehricke (cf. his Fig. 5-8 in [3]) for a two-stage rocket, a close analytic fit to which has been found to be:

$$C_{Do} = \begin{cases} 0.25 + 0.3773 \exp\left[-16.6426(M-1.14)^2\right] & (0 \le M \le 0.88889) \\ 0.6273 - 30.902\,|M-1.14|^{3.5} & (0.88889 < M \le 1.28) \\ 0.17913 + 0.5318/M & (1.28 < M) \end{cases} \tag{10}$$

$$C_{DL} = \begin{cases} 1.822 - 0.298 \cos\left[M\pi/1.086\right] & (0 \le 1.5) \\ 0.2818 + 3.2727/M & (1.5 < M) \end{cases} \tag{11}$$

$$\partial C_L/\partial \alpha = \begin{cases} & (0 \le M \le 1.5) \\ 0.8 + 6/M & (M > 1.5) \end{cases} \tag{12}$$

The graphs of Eq's (10) - (12) are shown in Fig. 4, along with a sketch of the type of rocket to which these coefficients apply. In this case, the reference value S in Eq's (7) and (8) is the cross-sectional area of the bottom

rocket stage. The dynamic pressure factor in Eq's (7) and (8) is seen to involve a competition between an atmospheric density ρ which is decreasing exponentially with altitude and a quadratic dependence on velocity which is increasing with altitude. The result of this in calculations is that aerodynamic forces are negligible above about 60 km, and peak somewhere around 10 km.

For the evaluation of thrust and aerodynamic forces an atmospheric model is needed, i.e., a specification of pressure ρ and density ρ vs altitude y in the region in which thrust variation and aerodynamic forces play a role (e.g., for $0 \le y < 100$ km). The temperature is related to p and ρ by the ideal gas law;

$$p = \rho \hat{R} T/M \tag{13}$$

where M is the mean molecular weight of one mole of atmospheric gas, and $\hat{R}$ is the gas constant given by $\hat{R} = 8.3143$ joules $\text{mole}^{-1}\text{deg}^{-1} = 8.3143 \times 10^7 \text{ergs mole}^{-1}\text{deg}^{-1}$. The speed of sound in Eq. (9) is calculated from

$$c^2 = 1.4\ p/\rho \tag{14}$$

where the factor 1.4 is the ratio of specific heat at constant pressure to the specific heat at constant volume for diatomic molecules, such as air. For use in the computer program it would be helpful to have an analytic expression of the atmospheric density and pressure altitude profiles. This is developed in Appendix A. Using the approach developed there and the numerical tabulation of Jastrow and Kyle [6], the following analytic forms for the atmospheric profile are obtained:

$$p = p_o (H/H_o)^{-1/\beta_o}$$

$$\rho = \rho_o\ [1-2(y-y_o)/(R+y_o)]^{-1}\ \ (H/H_o)^{-(1+\beta_o)/\beta_o}\ ,$$

where $H = H_o + \beta_o(y-y_o)$ $(y_i<y\leq y_f)$ (15)

Different parameters (ρ_o, p_o, H_o, β_o, y_o) are used for different altitude intervals [Y_i, Y_f], and these are given in Table 1. Use of these expressions is found to give a good fit of the numerical tabulation of Jastrow and Kyle [6].

TABLE I

Parameters for Analytic Atmospheric Profiles (See Eq. (15) of text)

Y_i (km)	Y_f (km)	Y_o (km)	ρ_o (kg./m.^{3})	P_o (N/m^2)	H_o (km)	β_o
0	10	10	4.176-01	2.614+04	6.4087	-2.0378-01
10	25	10	4.176-01	2.614+04	6.4087	2.0064-03
25	45	25	4.048-02	2.534+03	6.4388	6.9940-02
45	55	55	5.650-04	4.283+01	7.8707	2.4469-03
55	85	85	9.193-06	5.020-01	5.7235	-7.2257-02
85	90	85	9.193-06	5.020-01	5.7235	1.7711-03
90	100	90	3.842-06	2.098-01	5.7323	6.1446-02
100	120	100	6.642-07	4.005-02	6.3487	2.6826-01
120	140	120	3.613-08	4.324-03	10.7986	2.3350-01
140	D		0	0		

Integration of the Equations of Motions

Before going on to the actual integration of the equations of motion, further approximations used should be indicated. These have to do with the magnitude of the angle of attack α during the flight of the rocket.

As Eq. (2) indicates, while the rocket speed is relatively low during first stage motion, a particular value of α has a larger effect on the turning rate $\dot{\psi}$ than when the rocket speed is much greater in second or higher stage motion. Values of α in first stage motion needed to a cause a particular orbit injection value of ψ (e.g., ψ=0) have been found not to exceed a few degrees. On the other hand, the maximum value of α in higher stages has been found to be as high as tens of degrees in some cases, depending on payload weight and the required orbit injection altitude. At least while aerodynamic effects are important, however, it appears to be a good approximation to replace sin α by α, and the approximations of Eq's (7) and (8) are valid. Hence to a good approximation, Eq's (1) - (3) become

$$\dot{v} = -(K/r^2)\sin\psi - (D/M) + (T/M)\cos\alpha \qquad (16)$$

$$v\dot{\psi} = -\left[\frac{K}{r^2} - \frac{v^2}{r}\right]\cos\psi + \frac{1}{M}\left[T + \frac{\partial C_L}{\partial\alpha}\frac{\rho v^2}{2}S\right]\sin\alpha \qquad (17)$$

These are the equations of motion integrated in the computer program. They are considered valid for all rocket stages. Here D is given by Eq's (7), (10), and (11), and T is given by Eq. (6).

The procedure is to employ a particular profile for $\alpha(t)$ in the integration of Eq's (16) and (17). It has the form

$$\alpha(t) = \begin{cases} C\,(t-t_o)\left[1-\dfrac{(t-t_o)}{t_d}\right] & (t_o<t<t_o+t_d) \\ 0 & (t\le t_o \text{ or } t\ge t_o+t_d) \end{cases} \qquad (18)$$

It is shown in Fig. 5 as an inverted parabola where it is non-vanishing, with an extremum point at $t = t_o + t_d/2$ and associated value $\alpha_m = Ct_d/4$. It is a convenient profile to manipulate, and the results for orbit injection conditions have been found to be insensitive to variation of parameters in the profile. For example, a particular orbit injection altitude at $\psi = 0$ can be obtained for various C values by adjusting the t_d value, and the value of injection velocity is insensitive to this procedure (e.g., within 0.2%). The assumption of a particular form in Eq. (18) would be a major concern if this were not the case.

Beside having solutions for v(t) and ψ(t) in Eq's (16) and (17), it would also be of interest to find the altitude and ground range (cf. Fig. 1), given as solutions of

$$\dot{y} = v \sin \psi \tag{19}$$

$$\dot{x} = v R \cos \psi/(R+y) \tag{20}$$

The Eq's (16), (17), and (19), and (20) are a coupled set of ordinary first order differential equations in the time variable, and are therefore solvable by the Runge-Kutta method [7]. In explaining this method it will be convenient to rewrite these equations as

$$\dot{x}^{(\alpha)} = f^{(\alpha)}(t, \{x^{(\beta)}\}) \quad (\alpha,\beta=1,2,3,4), \tag{21}$$

where the superscript α in this case runs over the four variables v, ψ, y, and x, which are respectively represented by $x^{(1)}$, $x^{(2)}$, $x^{(3)}$, and $x^{(4)}$. Eq. (21) states that the time derivative of one of these variables is a particular function of the time (e.g., through the given explicit time dependences of $\alpha(t)$ in Eq. (18) and mass of the rocket M(t)) and of the four variables $\{x^{(\beta)}\}$ at that time. Actually, inspection of Eq's (16), (17), (19), and

(20) reveals no dependence of $f^{(\alpha)}$ on $x^{(4)} = x$ here. In the Runge-Kutta numerical method, the time scale is broken up into mesh points separated by a fixed interval width h. The integration is consecutively forward on the time scale, with the variable at the mesh point n determined from the known values at the mesh point n-1, as follows:

$$x_n^{(\alpha)} = x_{n-1}^{(\alpha)} + \frac{1}{6}\left(\Delta x_1^{(\alpha)} + 2\Delta x_2^{(\alpha)} + 2\Delta x_3^{(\alpha)} + \Delta x_4^{(\alpha)}\right) \quad (22)$$

where

$$\begin{aligned}
\Delta x_1^{(\alpha)} &= hf^{(\alpha)}\left(t_{n-1}, \{x_{n-1}^{(\beta)}\}\right) \\
\Delta x_2^{(\alpha)} &= hf^{(\alpha)}\left(t_{n-1} + h/2, \{x_{n-1}^{(\beta)} + \Delta x_1^{(\beta)}/2\}\right) \\
\Delta x_3^{(\alpha)} &= hf^{(\alpha)}\left(t_{n-1} + h/2, \{x_{n-1}^{(\beta)} + \Delta x_2^{(\beta)}/2\}\right) \\
\Delta x_4^{(\alpha)} &= hf^{(\alpha)}\left(t_{n-1} + h, \{x_{n-1}^{(\beta)} + \Delta x_3^{(\beta)}\}\right)
\end{aligned} \quad (23)$$

The procedure, according to these equations, is to first evaluate the four values of $\Delta x_1^{(\alpha)}$, which then enables the evaluation of the four values of $\Delta x_2^{(\alpha)}$. Next comes the evaluation of $\Delta x_3^{(\alpha)}$, and then $\Delta x_4^{(\alpha)}$. Finally, $x_n^{(\alpha)}$ is evaluated from Eq. (22). This is also the procedure for any number of variables (α, β, = 1,2, -----), not just the four of interest here, and the method is easily programmable.

The computer program for the rocket trajectory consists of a main program and ten subroutines. The general approach of the coordinating main program RALLY 6 is to take a trial $\alpha(t)$ function from Eq. (18), and then perform the numerical integration of Eq's (16), (17), (19), and (20) by the Runge-Kutta method outlined in Eq's (21)-(23). Based on the final rocket burnout values of $\psi(t)$ and $y(t)$ - i.e., the orbit injection values of these variables -

the tilt parameters in Eq. (18) are varied, and the process is repeated iteratively until the desired orbit injection conditions are obtained. Instead of varying tilt parameters the program also provides for the variation of coast times after powered stage burnouts, in order to obtain orbit injection conditions. The program is interactive with the user in that corrections are made by him during the run, but at some point the user can choose an option which makes the program self-correcting, i.e., the program iterates on the corrections by a Newton-Raphson method [7] until convergence to a specified (by the user) stage burnout heading is obtained within a specified tolerance.

The input and output options are very flexible, so that the program will handle a wide variety of rocket problems with ease. The Subroutines are: (1) MASS which evaluates the mass of the rocket in Eq's (16) and (17) from $M=M_o-\dot{M}t$, where the initial stage mass M_o and mass ejection rate $\dot{M}$ are known parameters for the rocket stage, and t is the burn duration in that stage; (2) TILT which evaluates angle of attack from Eq. (18); (3) DEN which evaluates density and pressure according to Eq. (15) and Table 1 and sound speed according to Eq. (14); (4) PSINCR which evaluates $\dot{\psi}$ in Eq. (17), which is needed for the Runge-Kutta integration in the main program; (5) VINCR which evalues $\dot{v}$ in Eq. (16); (6) ALT which evaluates $\dot{y}$ in Eq. (19); (7) GRRG which evaluates $\dot{x}$ in Eq. (20); (8) CNVG which computes the parameter corrections to tilt amplitude (C in Eq. (18)) or stage coast time in the self-correcting mode of operation of the program, and then tests for convergence; (9) STORE which either stores burnout values just computed for the present rocket stage to be used in calculations for the next stage, or recalls values stored from calculations on the preceding stage to be used in a repetition of the calculations for the present stage; and (10) FORCES which com-

putes thrust and aerodynamic forces in the equations of motion according to Eq's (5)-(12).

Detailed Description of the Computer Program and Its Use

In Appendix B is the listing of the program for the DEC-10 system, and in Appendix C is the almost identical listing of the program for the PDP-11/70 computer system. One of the few changes made arose from the way the two different computers evaluated cos ψ in Eq's (17) and (20). There was no problem with the DEC-10, but the PDP-11/70 made small errors in the evaluation of cos ψ near $\psi=\pi/2$ rad. As small as these errors were, they were compounded in the integration procedure, so that final errors in the burnout ground range and altitude amounted to a few km. or more. This was remedied in the PDP-11/70 program version by replacing cos ψ, wherever it occurred, by sin $(\pi/2-\psi)$. With respect to the DEC-10 listing it is seen that statements #12400 and 14900 were changed in the PDP-11/70 version in this fashion. There is also a difference in the treatment of the vacuum-to-sea level thrust ratio which is given a value VSLR(I) for each rocket stage in the DEC-10 version (cf. statements #120, 2010, and 4255 in Appendix B). This ratio is given as x in Eq's (5) and (6). Very often the effect of VSLR or x is negligible, except for the first stage, because rocket altitude in the upper stages is high enough that the atmosphere is essentially a vacuum, i.e., T takes on the value T_{vac} in Eq. (6). This is why VSLR is treated as a single parameter for the first stage in the PDP-11/70 version in Appendix C. The only other change was necessitated by the difference in printing output between the two systems. While the output is typed at the DEC-10 terminal on a paper roll as TYPE statements are encountered in the program, it is printed on a CR-tube display in the

PDP-11/70 system (which periodically erases itself) and subsequently printed on paper at a line printer terminal upon successful completion of the job. In the PDP-11/70 version of the program the IPR printing option variable in the DEC-10 version is replaced by ITY, and IPR becomes a printing option variable for the line printer. Otherwise, the programs are identical. Henceforth, the discussion will be confined to the DEC-10 version.

At this point the statements of the program listed in Appendix B will be itemized, and to help the reader follow the logic of the program, a flow chart of the main program (RALLY 6) is included in Fig. 6(a), (b), and (c). The first two DATA statements in Appendix B (#300 and 400) list the y_f and y_o values in Table 1, while the next one lists the values of ρ_o in g./cm.^{3}. The next data statement (#700) lists the values of $-2(R+y_o)^{-1}$ in km^{-1} (cf. Eq. (15)). Next come the values of H_o (in #1000), β_o (in #2000), and P_o in dynes/cm^2 (in #1350) from Table 1. This is followed by values for constants used in the program; sequentially, these are K in km^3/sec^2 (cf Eq's (1) and (2)), the radius of the earth R in km, pi, and the pressure at sea level (p(o) in Eq. (6)) in dynes/cm^2. After zeroing a few execution option parameters, the user reads in the number of rocket stages NS up to 6 (which can be changed to a higher number, if necessary, by changing statement #120) and SMZ (I), the initial mass for each rocket stage (in lbs.), which is defined as the mass of the rocket at the beginning of the stage. Periods of coasting can be included as separate stages. Next SMD (I) is read-in which is the rate of mass loss of each stage due to fuel consumption (in lbs/sec). This would be set to zero for a coasting stage. Then the vacuum thrust for each stage TH(I) is read (in lbs), which is the T_{vac} of Eq. (6). Following this, the program calls for the vacuum-to-sea level thrust ratio VSLR for

each stage (i.e., x in Eq. (5)), the duration time (in sec) for each stage TBD(I), the stage diameter DI(I) (in meters), and the number of time intervals NI (I) that each rocket stage is to be broken up into in the specification of the time mesh for the numerical integration. In statements #2605 - 2630, TBS(I) is computed, which is the time measured from launch in sec. at which the I'th rocket stage commences. Next the altitude YI, time TI, speed VI, heading PSI, altitude interval ID (associated with Table 1) and angle of attack AL are given values appropriate to launch conditions, and the program is then ready to enter the numerical integration procedure.

After testing a pair of parameters, which will be discussed later, the program calls for the specification of the angle of attack parameters in statements #3460 - 3550. Here CA TLZ, and TLD refer to C (in rad.), t_o (in sec), and t_d (in sec) in Eq. (18), and thus refer to the tilt amplitude, the time measured from launch at which tilt begins, and the duration of non-vanishing tilt, respectively. Now the user supplies the parameters which determine the various execution options in statement #3575:

(1) NT specifies the number of rocket stages to be integrated as in Eq's (21) - (23).

(2) IOP specifies what part of the program that control is returned to after the NT rocket stages have been integrated. IOP = 1 terminates the program. IOP=2 returns control to program statement 3 (PS3) where variables are given their launch values. IOP=3 returns control to PS69 which redefines the time mesh for the numerical integration. IOP = 4 returns control to PS25 which initiates the integration of rocket stages other than the first. IOP=5 returns control to PS2 which defines a new rocket problem.

(3) ICE specifies the number of rocket stages which

have previously been integrated.

(4) IPR specifies the extent of the printed output from the numerical integration of the NT rocket stages. IPR = 0 causes printout of the variables at all the mesh points. IPR ≥ 1 causes printout of only the final or burnout values of the variables. IPR = 0,1 signals the program that corrections to tilt or coast time parameters will be supplied by the user. IPR=2 signals that tilt amplitude or coast time corrections will be computed automatically by the program in its self-correcting mode until convergence has been obtained, i.e., when the burnout value of the heading (ψ) takes on a specified value within a specified tolerance.

(5) IBG specifies what the value of IOP will be after convergence has been obtained in the self-correcting mode of the program.

(6) JCV specifies what is to be corrected in the self-correcting mode of the program. JCV = 2 specifies that TBD(NS) is to be adjusted, i.e., the coast time of stage NS. JCV ≠ 2 specifies that CA is to be adjusted, i.e., the amplitude parameter for the angle of attack profile of a particular rocket stage.

(7) IMA specifies whether or not the payload weight is to be adjusted. This control is convenient for obtaining the dependence of orbit injection conditions on payload weight.IMA = 1 specifies that a payload weight increment is to be read-in after the program has obtained covergence in its self-correcting mode of operation.

(8) JNS specifies whether or not a new NS is to be read in for the integration procedure. JNS = 0 says not, but JNS ≠ 0 specifies that a new NS will be read in. A temporary NS value is used for the self-correcting mode of operation on parameters of an intermediate rocket stage when the NT stages being integrated do not include the final rocket stage.

If IPR = 2 the program enters its self-correcting mode of operation in the integration of NT rocket stages. The first part of this is the specification of the desired final result PSSS for the heading in the integration and the convergence within EPS of the parameter that is being corrected. These values are read in statements #3882 and 3885. The program then carries out the numerical integration of the Eq's of motion (16), (17), (19), and (20) as indicated

in Eq's (21) - (23), and with the help of the subroutines discussed at the end of the preceding section. The reader will have little difficulty in following this. While some of the parameters were read in CGS or English units, everything is converted to MKS units in the numerical integration with the exception that distance units are km. in the final results.

Beginning with statement #8010 the program enters a series of tests which are designed to correctly set the parameter ICD for the subroutine STORE, which is called in statement #8050. If ICD = 1, STORE will store the trajectory values just computed, to be used as initial values for integration of the next stage. If ICD = 2, STORE will recall the initial values for the integration just computed, so that this integration can be repeated to get a heading which is closer to that desired. The rest of the main program, which is indicated by the flow chart, is mostly concerned with obtaining convergence in the self-correcting mode of operation. Convergence is on the tilt amplitude CA or on the coast time of the stage NS (i.e., TBD(NS)), as determined by JCV. As previously stated, the sub-routine CNVG computes the corrections to these parameters in the self-correcting mode (see statement #8655). The parameter ICVG signals whether or not convergence has been obtained. ICVG = 0 means it has not; ICVG = 1,2 means it has.

Perhaps the best way to understand how the program works and how to use it is to trace through the flow chart in Fig. 6 with data on sample problems. For this purpose, consideration is directed to a two-stage rocket with four stages of motion. The author would like to make the distinction between rocket stages and stages of motion. The program is solely concerned with the latter usage, and the subsequent discussion here is too. The first stage of motion is powered, the second stage is defined as a

coasting stage, the third stage is powered, and the fourth stage is a coasting stage for the payload. It is assumed that all the rocket data have been read in, so that the point in the program has been reached where the angle of attack data are to be read-in, i.e., at statement #3500 in the program. The solutions of four problems for this rocket are of interest, and these are given below.

Problem 1: The right amount of tilt must be put in the first stage so that the rocket will be injected into orbit at zero heading (i.e., $\psi = 0$ as the final solution of the equations of motion). Hence, the second through fourth stages fly in a gravity-turn ($\alpha = 0$ in Fig. 1) trajectory. The user must supply the data shown in Table 2, which also shows the statement number that demands the data, along with occasional comments. The reader should trace the path of logic through the flow chart which is dictated by the data. The initial reading of the tilt or angle of attack data will result in a heading which is non-zero in the printed output. This suggests a new CA value which is read-in (see Table 2), and the change of IPR from 1 to 2 in the next data input will cause the program to go into its self-correcting mode of operation. From then on the program iterates on CA until corrections to CA become less than 0.000002. This will typically take only a few seconds of actual runtime on the DEC-10 system. Then the complete launch trajectory is printed out at every time mesh point as the converged solution. Note that the user supplies two trial values of CA before the correction procedure is made automatic. Two such values are required in the calculation of the correction in Subroutine CNVG. Program control then returns to PS3, where now ICVG = 0 and IPR = 0, and variables are initialized at their launch values. The program is now ready to solve the second problem.

TABLE II

Data Required for Problem 1 in Text

DATA SUPPLIED	STATEMENT NO. (#)	COMMENTS
CA, TLZ,TLD	3500, 3505, 3510	CA<0
42012	3626	JCV=IMA-JNS=0
CA,TLZ,TLD	3500, 3505, 3510	A new CA
42022	3626	
0.	3882	Final ψ
0.000002	3885	Sample tolerance

TABLE III

Data Required for Problem 2 in the Text

DATA SUPPLIED	STATEMENT NO. (#)	COMMENTS
CA, TLZ, TLD	3500, 3505, 3510	First stage values
24002	3626	
CA, TLZ, TLD	3500, 3505, 3510	Third stage values
24212	3626	
CA, TLZ, TLD	3500, 3505, 3510	A new CA - third stage
24222	3626	
0.	3882	
0.000002	3885	

Problem 2: With a specified tilt in the first stage, the tilt in the third stage must be adjusted so that the final orbit injection heading is zero. Table 3 specifies the data for this problem. The first stage tilt value is read-in, the first two stages are integrated, with the trajectory values being printed out, and the final results for the variables are stored for use as initial values in the integration of subsequent stages. Then trial tilt parameters for the third stage are read-in, and the injection conditions are calculated and printed out as a result of the integration of the third and fourth stages. Based on this, a new CA for the third stage is entered by the user along with the other tilt parameters. Then the program is put into its self-correcting mode, iteration on CA for the third stage ensues, and finally the converged launch trajectory is printed out. Control is then returned to PS3, where the variables are initialized at their launch values. The program is now ready to solve the third problem.

Problem 3: With specified tilt functions in the first and third rocket stages and a specified coast time for the second stage (TBD(2)), it is desired to adjust the coast time of the fourth stage to obtain $\psi = 0$ as the orbit injection heading. The data supplied by the user is shown in Table 4. By putting JCV = 2, the correction procedure is now on TBD(4). It proceeds analagously to that on CA. After converging to the correct coast time for the fourth stage, program control reverts back to PS3, where variables are initialized at their launch values again, and the fourth problem is now ready to be solved.

Problem 4: With specified tilt functions in the first and third rocket stages, it is desired to adjust the coast

TABLE IV

Data Required for Problem 3 in Text

DATA SUPPLIED	STATEMENT NO. (#)	COMMENTS
CA, TLZ, TLD	3500, 3505, 3510	First stage
24012	3626	
CA, TLZ, TLD	3500, 3505, 3510	Third stage
14212	3626	
0.	3500	These tilt
0.	3505	parameters play
0.	3510	no role (no thrust)
143122	3626	
TLS	8687	New coast time fourth stage
0., 0., 0.	3500, 3505, 3510	
143222	3626	
0.	3882	
.000002	3885	

TABLE V

Data Supplied by User for Problem 4 in Text

DATA SUPPLIED	STATEMENT NO. (#)	COMMENTS
CA, TLZ, TLD	3500, 3505, 3510	First stage
14001	3626	
0., 0., 0.	3500, 3505, 3510	Second stage
14114201	3626	Coast time iteration
2	3845	Temporary NS
TLS	8687	New coast time
0.,0.,0.	3500, 3505, 3510	Irrelevant tilt again
141242	3626	Self-correct coast time
pSSS	3882	ψ_2
.000002	3885	Tolerance
Ca, TLZ, TLD	3500, 3505, 3510	Third stage
14204001	3626	
4	3845	Restore original NS
0., 0., 0.	3500, 3505, 3510	Fourth stage
143122	3626	
TLS	8687	New coast time
0., 0., 0.	3500, 3505, 3510	Irrelevant tilt again
143222	3626	
0.	3882	Orbit injection heading
.000002	3885	

time of the second stage to obtain a particular heading ψ_2 at the end of the second stage, and then to adjust the coast time of the fourth stage to obtain an orbit injection heading $\psi=0$. The data supplied by the user is shown in Table 5. The rocket launch trajectory is printed out for all four stages in the course of the program.

As demonstrated by the examples discussed, the program is flexible enough to handle a wide variety of rocket problems. Whatever rocket problem the user is dealing with, it seems advisable to check the data to be supplied to the program by tracing its path through the flow chart.

<u>Example: A Hypothetical 2-stage Launch Vehicle</u>

For typical input data the numerical output from a specific example is obtained next, and its use in computations is demonstrated. As in the preceding section, the rocket in this example has four stages of motion, the first and third being powered and the second and fourth being coasting stages. One might typically be given the rocket data of Table 6. It is a simple matter to calculate the input parameters for the program from Table 6 with the help of the following equations:

$$
\begin{aligned}
&SMZ = M_o \\
&SMD = T_{vac}/I_{sp} \\
&TH = T_{vac} \\
&VSLR = x \\
&(TBD)(SMD) = M_p
\end{aligned}
\qquad (24)
$$

The left-hand sides of these equations are written in the notation of the computer program (Appendix B), while the right-hand sides are written in the notation of Table 6. In Appendix D the solutions of Problem 1 and Problem 2 of the preceding section are obtained on a remote time-sharing DEC-10 terminal for the hypothetical rocket of

TABLE VI

Rocket Data for a Hypothetical 2-Stage Launch Vehicle

PARAMETER	STAGE	FIRST	SECOND
Ignition Wt. $\equiv M_o$	(lbs)	400000	120000
Stage Wt. $\equiv M_s$	(lbs)	274000	110000
Total Propellant Wt. $\equiv M_p$	(lbs)	264000	105000
Vac. Spec. Impulse $\equiv I_{sp}$	(sec)	302.3	314.3
Vac, Thrust $\equiv T_{vac}$	(lbs)	665000	220000
Vac.Sea level Thrust Ratio $\equiv x$		1.15	———
Stage diameter	(ft)	10	10

Table 6. The user enters data for the rocket problem when the program prompts him to do so with an ENTER---- message. The value 10 seconds is arbitrarily used for the coast times of the second and fourth stages of the rocket motion. This might, for example, allow enough time between burnout of one powered stage and ignition of the next powered stage to take care of jettisoning launch vehicle vestiges. In this particular exercise the user is trying to obtain the orbit injection velocity at zero heading (in this case, the perigee of the orbit) and at an altitude of 100 nmi. = 185.2 km. In order to find this, the user plots the three output values altitude vs. velocity, as in Fig. 7 (x - marks), and then interpolates (to o-mark) to find an injection velocity $v' \approx 7.833$ Km/sec. Incidentally, the total computer runtime for the output in Appendix D was 26 seconds.

The value just found for the orbit injection velocity at 100 nmi. altitude is in a stationary earth approximation, as discussed in the Introduction. The earth's rotation is easy to account for [1] in separate computations. To do this, one notes first that the total ground range covered in this example is about 670km., which is small enough that a flat earth approximation [1] is in order. Then the actual orbit injection conditions for the rocket, including earth's rotation, are given by

$$v = [v'^2 + 2v'v_o \sin a_o + v_o^2]^{\frac{1}{2}}$$

$$\psi = 0^o \qquad (25)$$

$$\tan a = \frac{v' \sin a_o + v_o}{v' \cos a_o}$$

where

$$v_o \equiv \omega_E R\, [1 + (y_1/R]\cos L_o$$

Here, v' is the injection velocity in the stationary earth

approximation, v_o is the effective initial velocity in an easterly direction imparted by earth's rotation, a_o is the launch azimuth angle as seen by an earth observer, which is established shortly after launch by suitable pitch and yaw rotations, and a is the actual azimuthal angle at orbit injection as seen by an inertial observer. These azimuthal angles are measured clockwise from the local north direction to the direction of motion projected on the local horizontal plane. In the expression for v_o, ω_E is the angular rotation rate of the earth, R is its radius, y_1 is the rocket altitude after first stage burnout, and L_o is the latitude of the launch site. The orbit inclination angle i, which is the angle made by the normal to the orbital plane (determined from the right-hand rule by curling one's fingers in the direction of motion of the satellite) with the earth's rotation axis, is determined from the preceding parameters by

$$\cos i = \sin a \cos L_o \quad , \tag{26}$$

where

$$o \le a \le 2\pi \quad \text{and} \quad |L_o| \le i \le \pi - |L_o|$$

in radian measure. By way of illustration it is supposed that the rocket in the example is launched from Cape Kennedy, for which $L_o = 28.5^o$. One has that $v' = 7.833$Km/sec, ω_E = 7.292116 rad/sec., R = 6378.145km., $y_1 \approx 64$km., and a_o is variable in the example. Fig. 8 is a plot of a and i vs. a_o and in Fig. 9 is a plot of v vs. i. This is as much information as is needed for the launch problem example. It will be seen from these figures and Fig. 3 that, as expected, the apogee altitude will be maximum when the launch direction is due east ($a_o = 90^o$); then it is about 1075nmi. On the other hand, for launch directions in the range $190^o \le a_o \le 350^o$ there is a question of the stability of the parking orbit, since in this range orbit altitudes dip below 100nmi.

DISCUSSION

The program developed here was employed in calculations on large launch vehicles. Curves of the type of Fig. 7 were found for the injection conditions in the stationary earth approximation. Earth's rotation effects were subsequently included in the manner described previously. The results were compared to corresponding calculations which leave out explicit consideration of thrust increase and drag effects, but rather assume that they cancel in an approximation delineated by Ruppe [4]. A similar approximation was used by Ehricke [3]. The curves based on this approximation yielded injection speeds at a given altitude 2-3 % less than those calculated with the present program. For very large rockets the thrust increase effect outweighs the drag effect by about this amount. Because of the sensitive dependence of apogee altitude on perigee injection speed for eccentric orbits, it was concluded that explicit inclusion of thrust increase and drag effects was necessary. It was also possible to calculate results with the present program for a large launch vehicle which had been independently calculated elsewhere. These were results of the nature of Fig. 9, and the agreement between the two sets of calculations was within 0.05%. While this kind of agreement is probably fortuitous, it is also somewhat encouraging.

Realistically, one would expect the present program to yield information about large, unfinned rocket systems, consistent with the data base to within 1% for the injection speed. The neglect of angle of thrust (β in Fig. 2) is supposed to be a good approximation for large rockets [3, 4], and a particular form for the angle of attack profile is apparently unimportant for these systems. A variation of parameters in the profile used here produced $\lesssim$ 0.2% errors. Beyond this, one can argue that small errors

in the aerodynamic coefficients or atmospheric model are not important, because the total contribution of aerodynamic effects is relatively small for large rockets [4]. Another consideration is that significant improvements on the present program are probably not possible for many rocket problems. Data for such a problem are often comparable with that of Table 6. Further refinements on the approximations would require more data than is given. One would need, for example, drag and lift coefficients specific to the rocket system, details of the angle of attack or thrust angle profiles, the time history of thrust and mass variations, and data on the atmospheric conditions. Furthermore, uncertainties in rocket parameters in the data base can and often do amount to a few percent, and this is therefore an inherent limitation to accuracy.

The present program is certainly flexible enough to handle a large variety of rocket problems. From the preceding discussion it appears to be entirely satisfactory for most launch system analyses, particularly those which involve large, unfinned rockets.

ACKNOWLEDGEMENT

The author would like to thank Mr. Frank D. Clarke who entered the program and adapted it to the DEC-10 and PDP-11/70 systems. In addition to his help in the debugging and in numerous revisions of the program, he also assisted in obtaining results for some rocket problems. The author also appreciates assistance from Dr. John N. Hayes on the manuscript.

REFERENCES

1. M. H. Reilly, NRL Report 8237, "Equations of Powered Rocket Ascent and Orbit Trajectory" (April 1978).

2. K. J. Ball and G. F. Osborne, "Space Vehicle Dynamics" (Oxford University Press, New York, 1967).

3. K. A. Ehricke, "Space Flight II Dynamics", (D. Van Nostrand Co., New York, 1962).

4. H. O. Ruppe, "Introduction to Astronautics", Vol. 1, (Academic Press, New York, 1966).

5. J. M. J. Kooy, "Dynamics of Controlled Rocket Launching pp 31-69 in Dynamics of Rockets and Satellites, ed. by G. V. Groves (North-Holland Publ. Co., Amsterdam, 1965)

6. R. Jastrow and H. Kyle, Sec. 2.1 in Handbook of Astronautical Engineering, ed. by H. H. Koelle (McGraw-Hill, New York, 1961.

7. See, e.g., B. Carnahan, H. A. Luther, and J. O. Wilkes, Applied Numerical Methods (John Wiley & Sons, New York 1969).

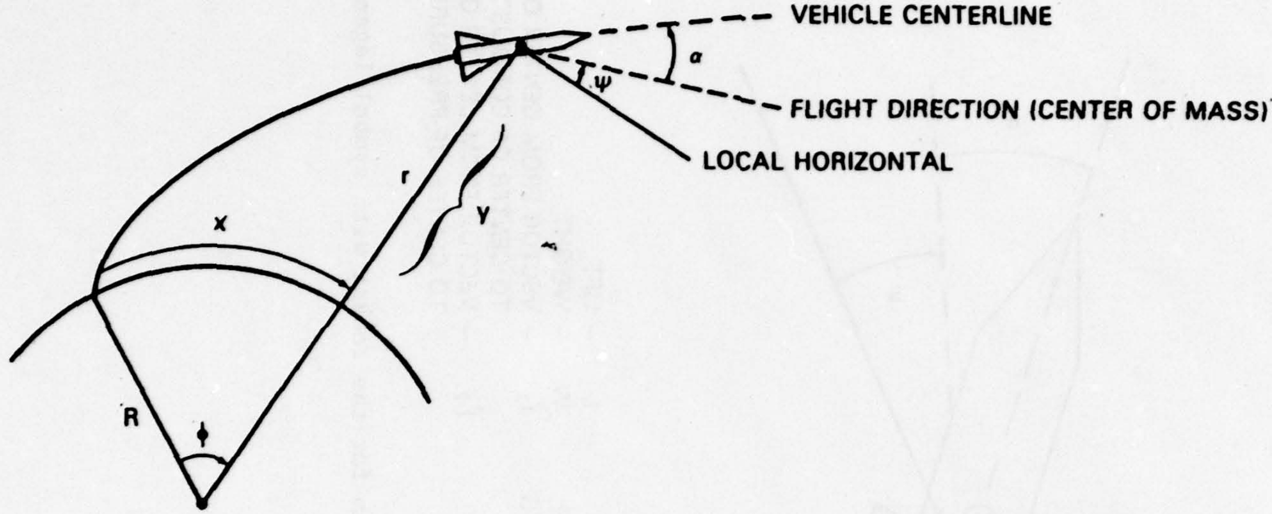
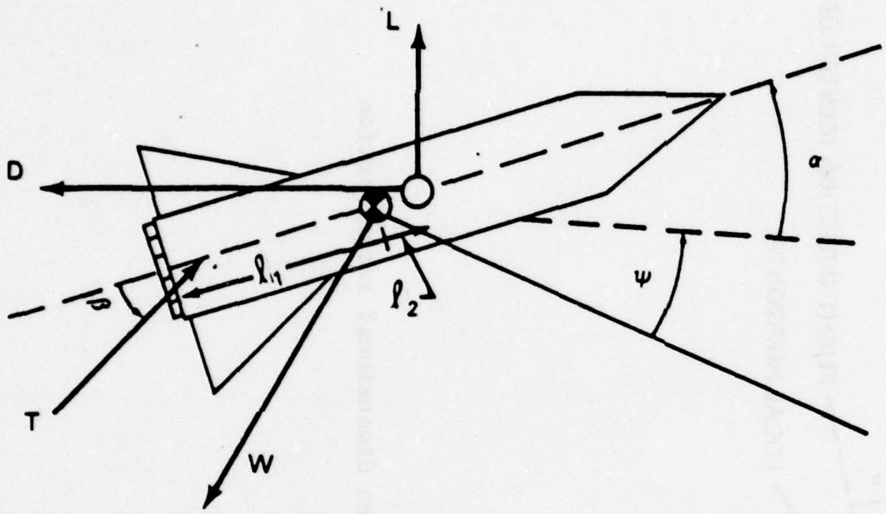


Fig. 1 - Coordinate system for two dimensional rocket motion



⊗	– CENTER OF GRAVITY		L	– LIFT	
O	– CENTER OF PRESSURE		W	– WEIGHT	
ψ	– FLIGHT PATH HEADING		ℓ_1	– VECTOR FROM CENTER OF GRAVITY TO CENTER OF COMBUSTION	
α	– ANGLE OF ATTACK				
T	– THRUST FORCE		ℓ_2	– VECTOR FROM CENTER OF GRAVITY TO CENTER OF PRESSURE	
β	– ANGLE OF THRUST				
D	– DRAG				

Fig. 2 - Force diagram for the rocket with symbol legend

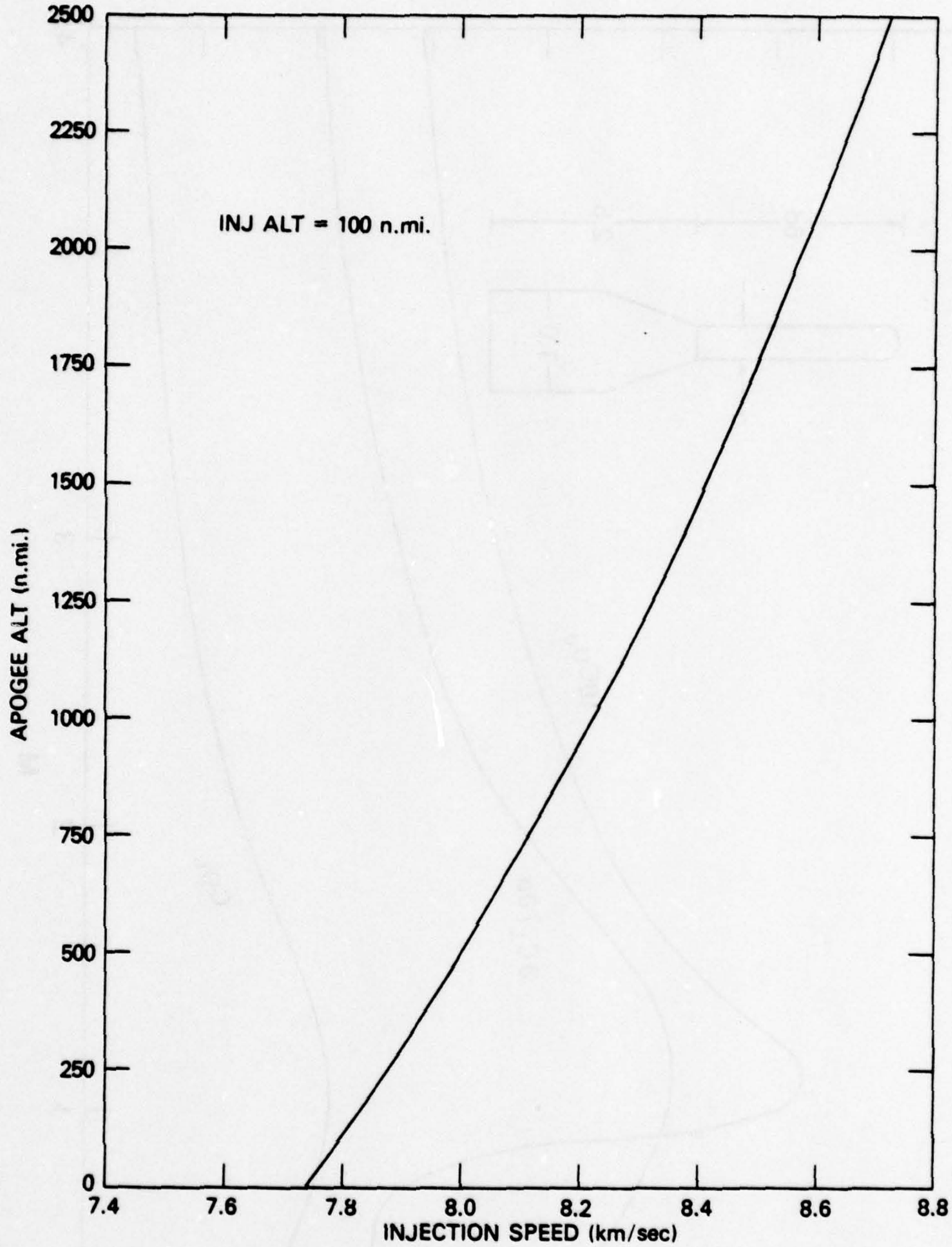


Fig. 3 - Apogee altitude dependence on perigee injection speed at an altitude of 100 nmi

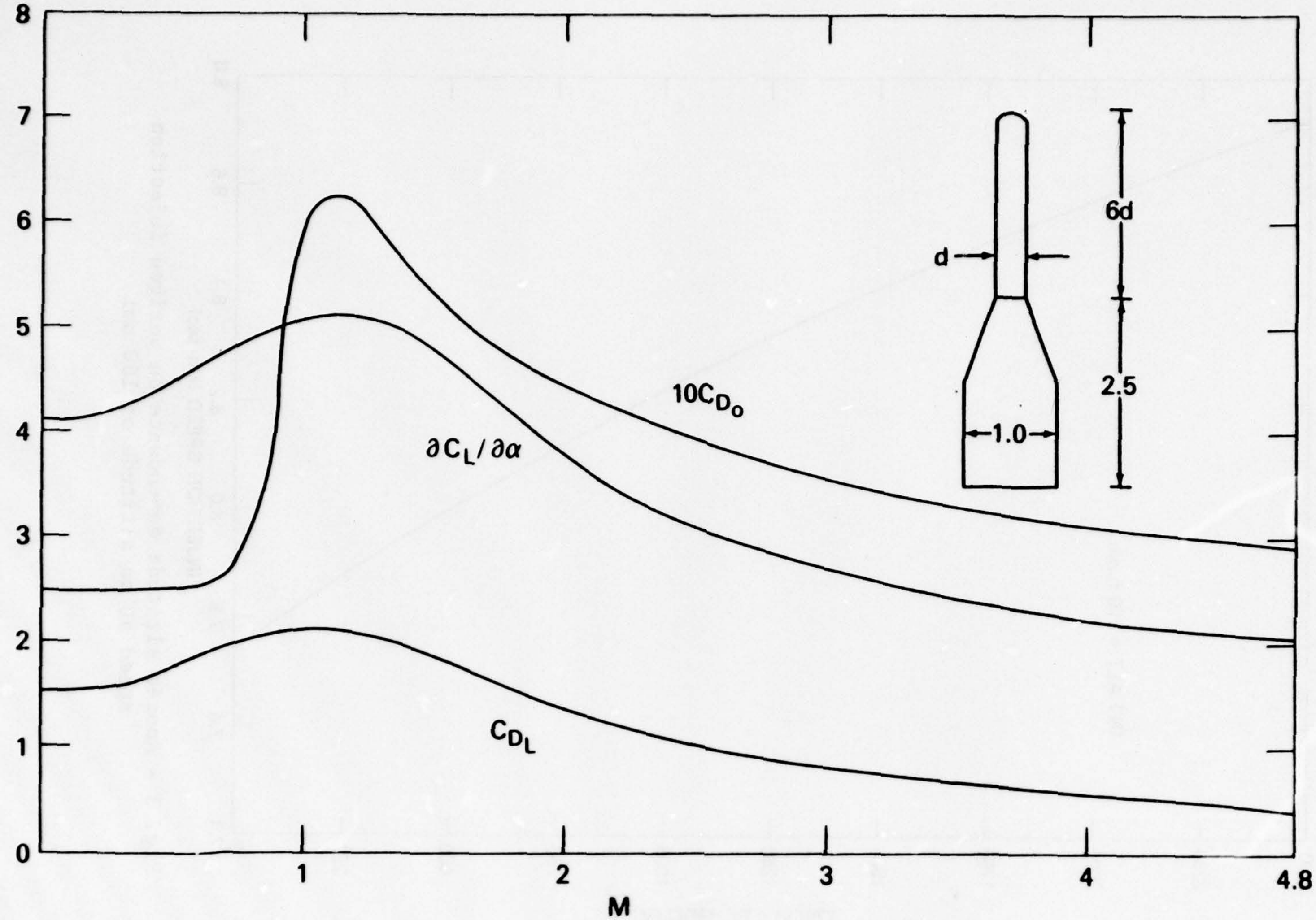


Fig. 4 - Aerodynamic coefficients dependence on local mach number for the indicated two-stage rocket

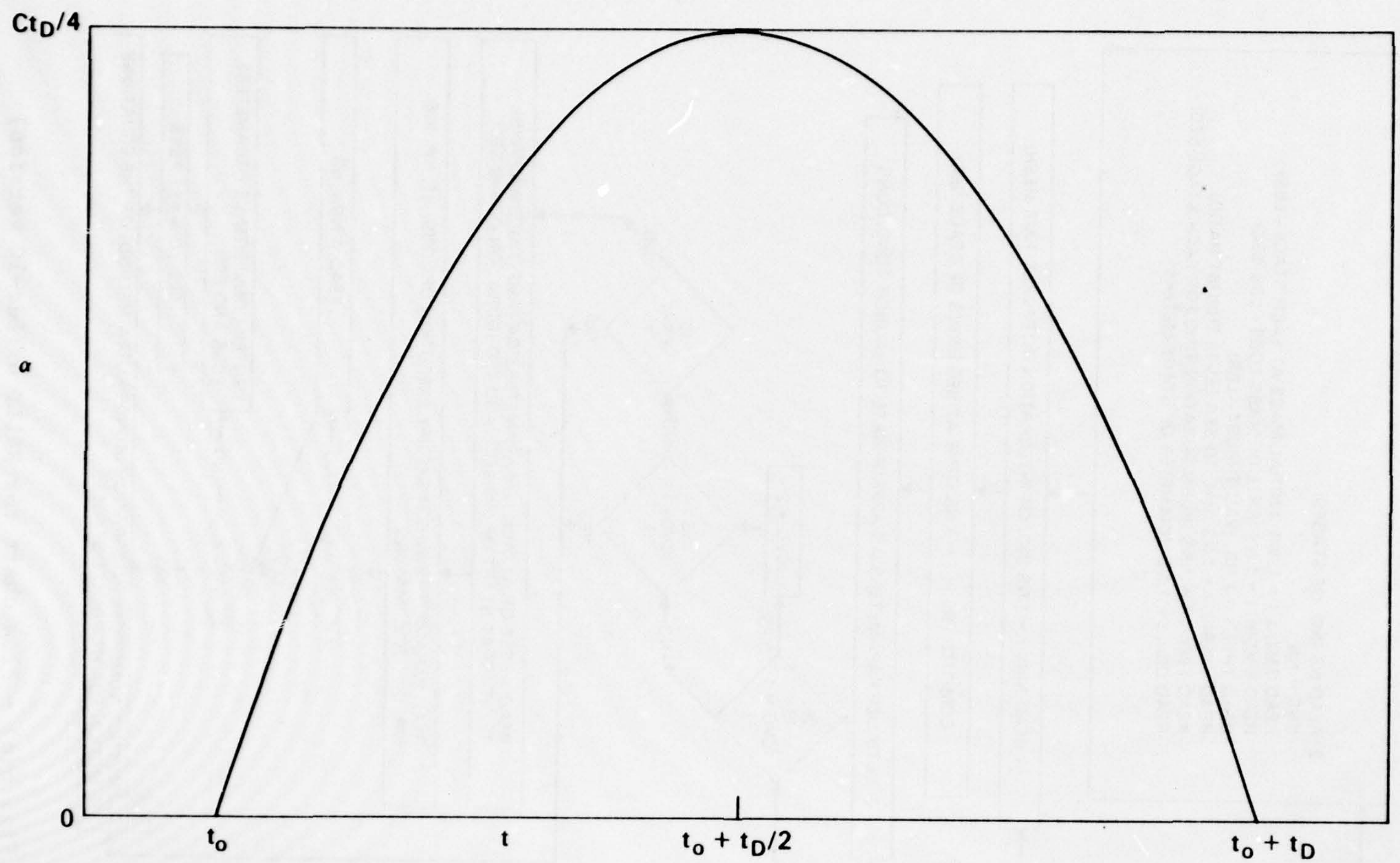


Fig. 5 - Angle of attack time profile used for computer program.

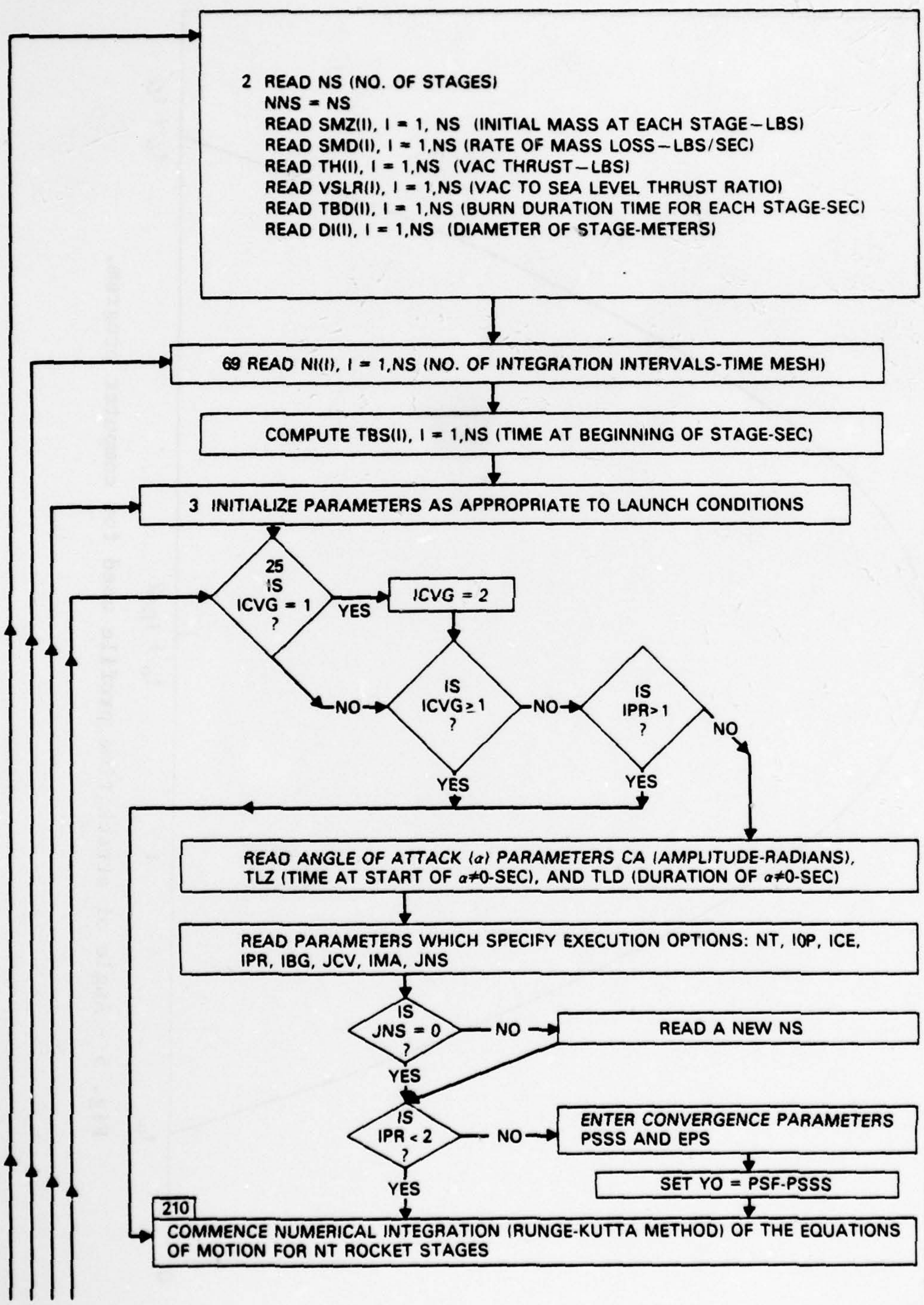


Fig. 6(a) - Flow chart for RALLY 6 (Dec-10 version)

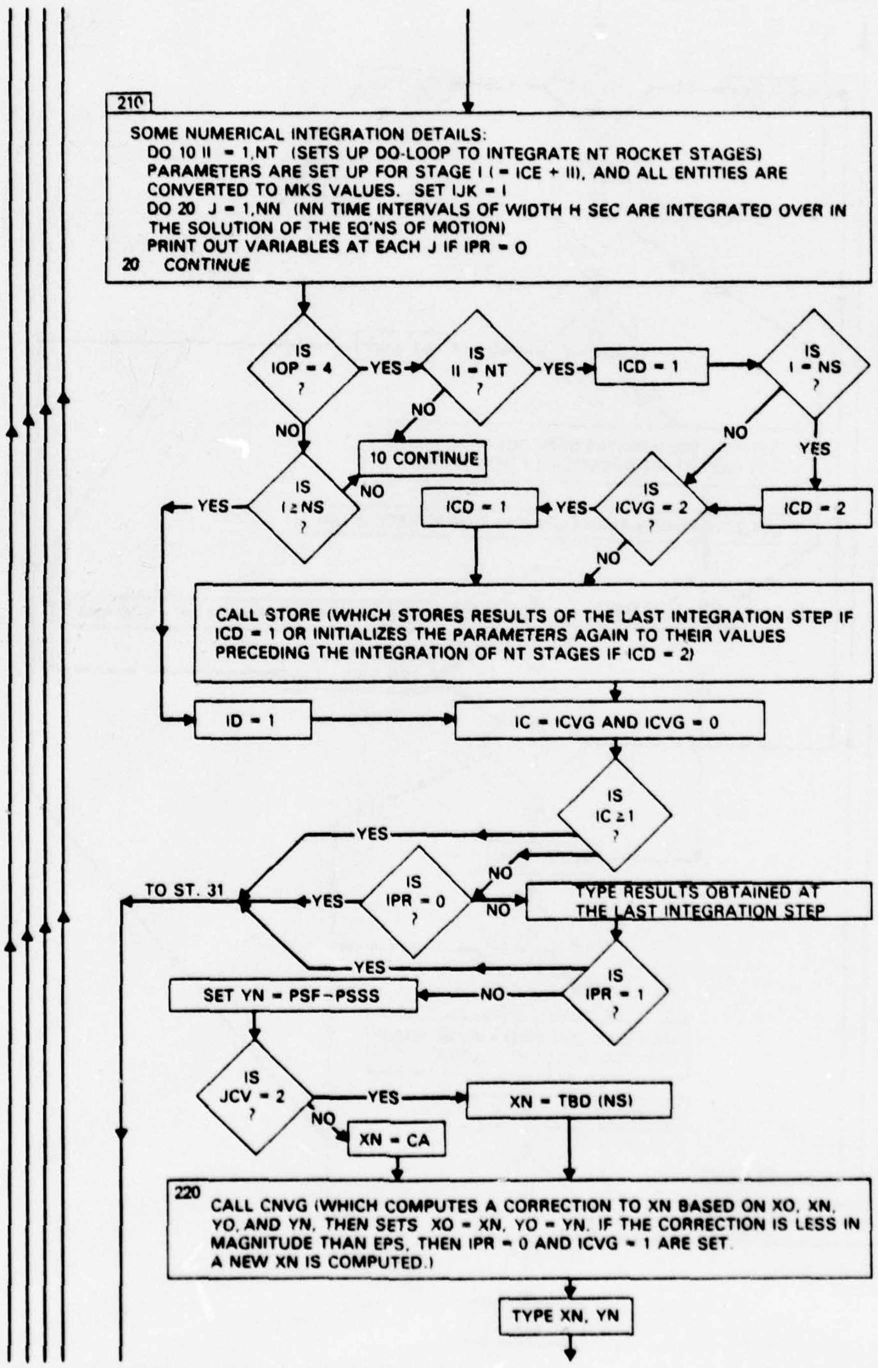


Fig. 6(b) - Flow chart for RALLY 6 (Dec-10 version)

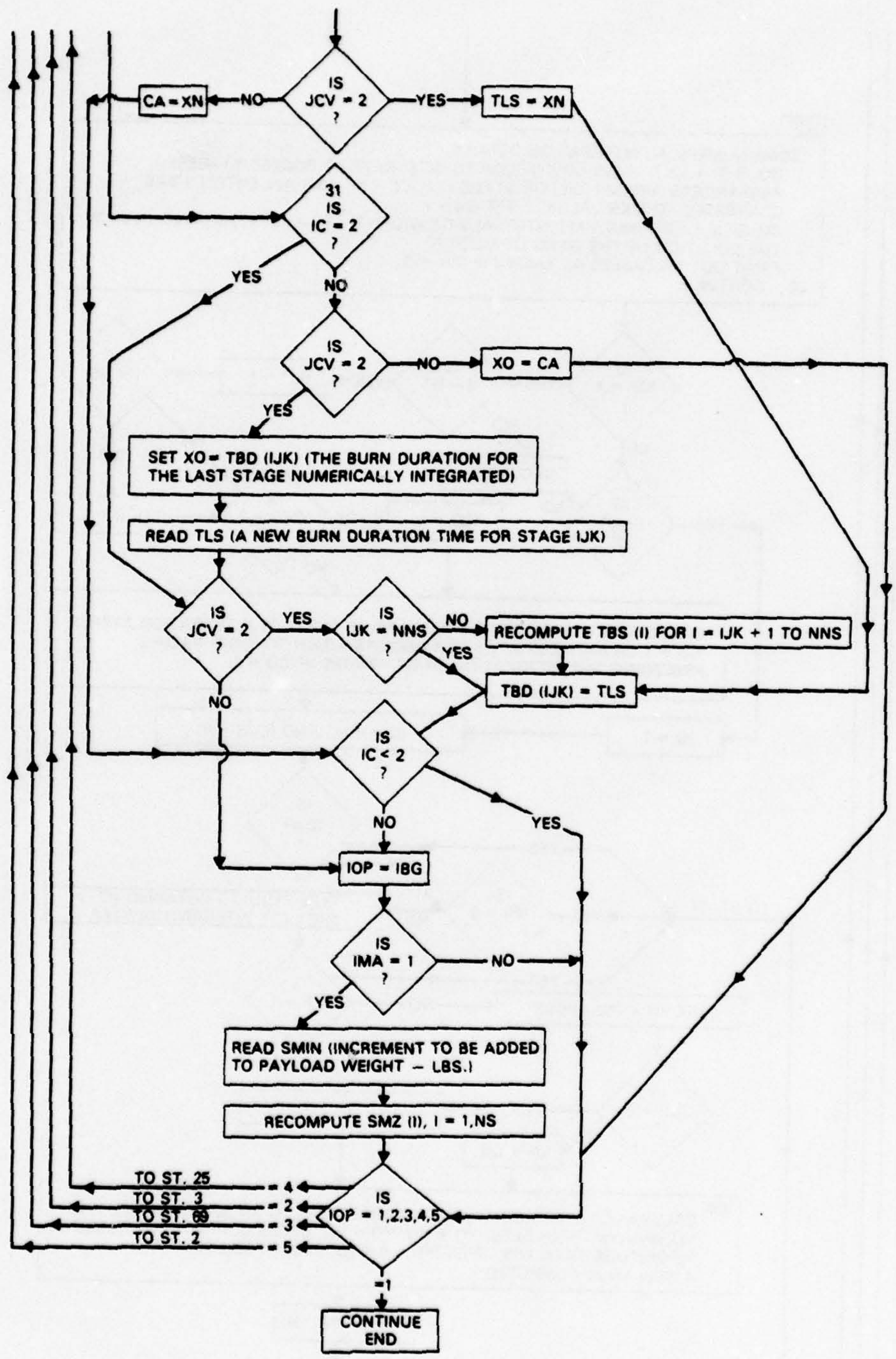


Fig. 6(c) - Flow chart for RALLY 6 (Dec-10 version)

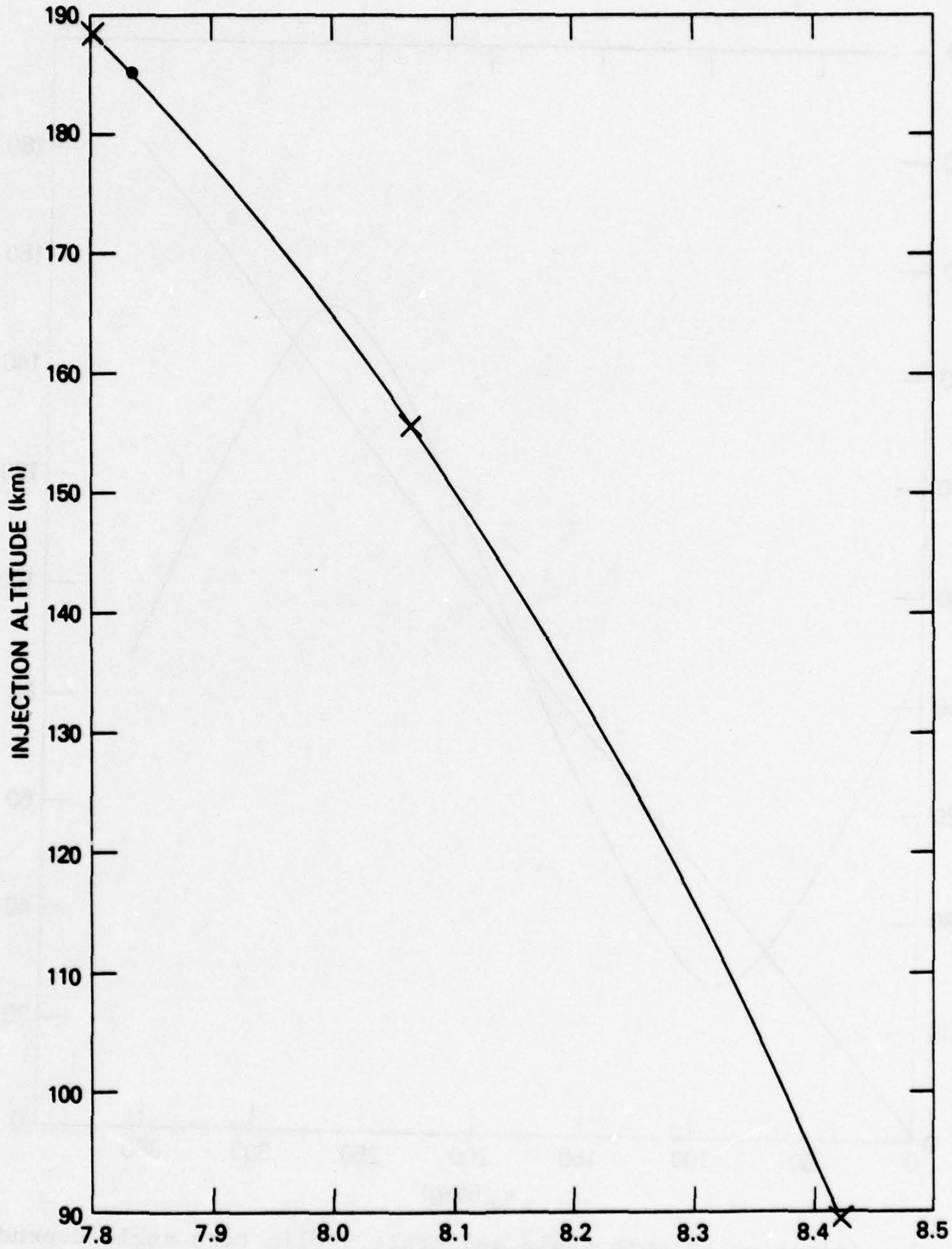


Fig. 7 - Altitude vs. speed at orbit injection for the example rocket (see text) in stationary earth model

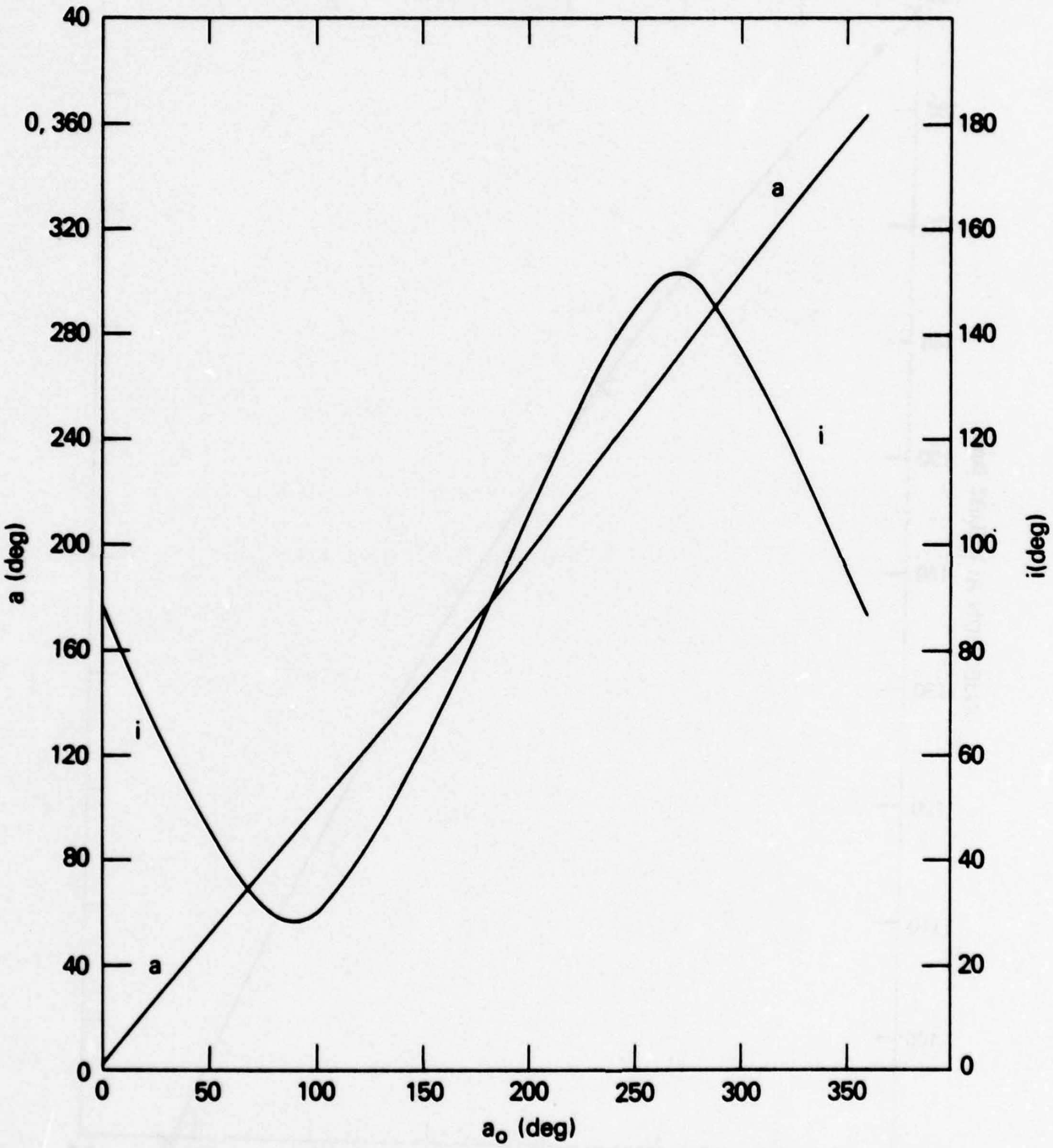


Fig. 8 - Injection azimuth angle and orbit inclination angle dependence on launch azimuthal angle for the example rocket (see text)

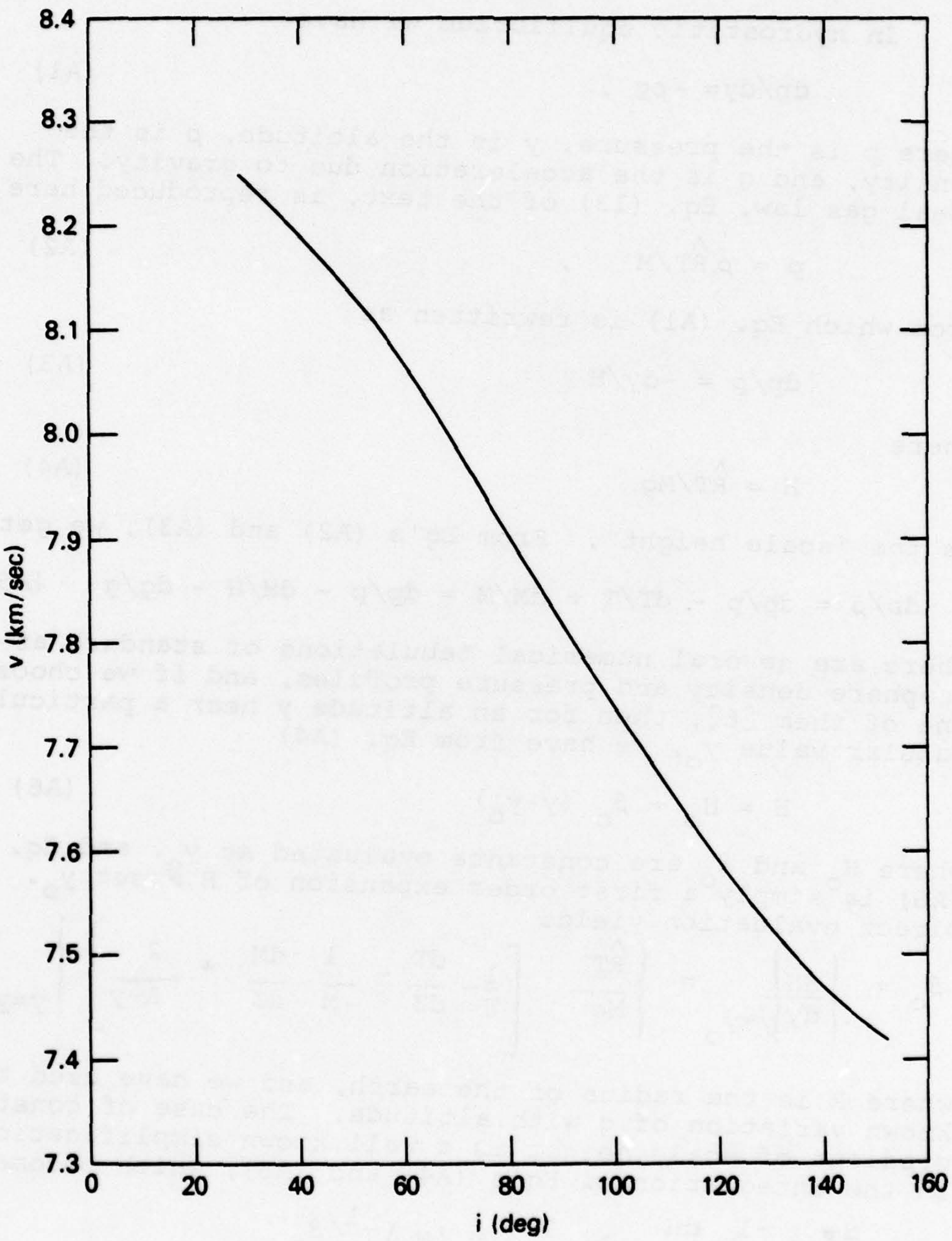


Fig. 9 - Injection speed dependence on orbit inclination angle for the example rocket (see text)

APPENDIX A

Analytic Atmospheric Density and Pressure Profiles

In hydrostatic equilibrium we have

$$dp/dy = -\rho g \ , \tag{A1}$$

where p is the pressure, y is the altitude, ρ is the density, and g is the acceleration due to gravity. The ideal gas law, Eq. (13) of the text, is reproduced here as

$$p = \rho \hat{R}T/M \ , \tag{A2}$$

from which Eq. (A1) is rewritten as

$$dp/p = -dy/H \tag{A3}$$

where

$$H = \hat{R}T/Mg \tag{A4}$$

is the "scale height". From Eq's (A2) and (A3), we get

$$d\rho/\rho = dp/p - dT/T + dM/M = dp/p - dH/H - dg/g \tag{A5}$$

There are several numerical tabulations of standard atmosphere density and pressure profiles, and if we choose one of them [6], then for an altitude y near a particular tabular value y_o, we have from Eq. (A4)

$$H = H_o + \beta_o \ (y-y_o) \tag{A6}$$

where H_o and β_o are constants evaluated at y_o, and Eq. (A6) is simply a first order expansion of H about y_o. Direct evaluation yields

$$\beta_o = \left\{ \frac{dH}{dy} \right\}_{y=y_o} = \left\{ \frac{\hat{R}T}{Mg} \left[\frac{1}{T} \frac{dT}{dZ} - \frac{1}{M} \frac{dM}{dZ} + \frac{2}{R+y} \right] \right\}_{y=y_o} \tag{A7}$$

where R is the radius of the earth, and we have used the known variation of g with altitude. The case of constant gradient of scale height is a well known simplification in the integration of Eq's (A4) and (A5), which become

$$\frac{d\rho}{\rho} = \frac{-1}{\beta_o} \frac{dH}{H} \Rightarrow \frac{p}{p_o} = \left(\frac{H}{H_o}\right)^{-1/\beta_o} \tag{A8}$$

$$\frac{d\rho}{\rho} + \frac{dg}{g} = \frac{d(\rho g)}{\rho g} = -\left(\frac{\beta_o+1}{\beta_o}\right)\frac{dH}{H}, \quad \frac{\rho g}{\rho_o g_o} = \left(\frac{H}{H_o}\right)^{-(1+\beta_o)/\beta_o} \tag{A9}$$

These equations are thus interpolation formulae, which when used in conjunction with some of the values in the numerical tabulation [6], become the desired analytic expression of atmospheric density and pressure (see text).

APPENDIX B
DEC-10 Version of RALLY 6

```
.TYPE RALLY6.FOR
           COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
           COMMON/B1/YZ(9),YD(9),RHZ(9),ALZ(9),HZ(9),BT(9),PR(9)
           COMMON/C1/SMZ(6),SMD(6),TBS(6),TBD(6),DI(6),TH(6),NI(6)
           DATA(YD(I),I=1,9)/10.,25.,45.,55.,85.,90.,100.,120.,140.
/
           DATA(YZ(I),I=1,9)/10.,10.,25.,55.,85.,85.,90.,100.,120./
           DATA(RHZ(I),I=1,9)/4.176E-04,4.176E-04,4.048E-05,5.650E-
07,
           19.193E-09,9.193E-09,3.842E-09,6.642E-10,3.613E-11/
           DATA (ALZ(I),I=1,9)/-3.1308E-04,-3.1308E-04,-3.1235E-04,
           1-3.1039E-04,-3.0945E-04,-3.0945E-04,-3.0921E-04,-3.0873
E-04,
           2-3.0779E-04/
           DATA(HZ(I),I=1,9)/6.4087,6.4087,6.4393,7.8707,5.7235,5.7
235,
           15.7323,6.3487,10.7986/
           DATA(BT(I),I=1,9)/-2.0378E-01,2.0064E-03,6.994E-02,2.446
9E-03,
           1-7.2257E-02,1.7711E-03,6.1446E-02,2.6826E-01,2.3350E-01
/
           DATA (PR(I),I=1,9)/2.614E05,2.614E05,2.534E04,4.283E02,
           15.02,5.02,2.098,4.005E-01,4.324E-02/
           GC=3.986012E05
           R=6378.145
           PI=3.141592654
           PRZ=1.013255E05
           ICVS=0
           IPR=0
    2      TYPE 120
    120    FORMAT (' ENTER NS')
           READ (5,100) NS
           NNS=NS
           TYPE 130,NS
           READ (5,200) (SMZ(I),I=1,NS)
           TYPE 140,NS
    130    FORMAT (' ENTER ',I1,' VALUES FOR SMZ')
           READ (5,200) (SMD(I),I=1,NS)
           TYPE 150,NS
    140    FORMAT (' ENTER ',I1,' VALUES FOR SMD')
           READ (5,200) (TH(I),I=1,NS)
           TYPE 152
    152    FORMAT (' ENTER VSLR')
           READ (5,200) VSLR
           TYPE 160,NS
    150    FORMAT (' ENTER ',I1,' VALUES FOR VAC. TH')
           READ (5,200) (TBD(I),I=1,NS)
           TYPE 170,NS
    160    FORMAT (' ENTER ',I1,' VALUES FOR TBD')
           READ (5,200) (DI(I),I=1,NS)
    69     TYPE 180,NS
    170    FORMAT (' ENTER ',I1,' VALUES FOR DI')
           READ (5,300) (NI(I),I=1,NS)
    180    FORMAT (' ENTER ',I1,' VALUES FOR NI')
    100    FORMAT (I1)
    200    FORMAT (F10.0)
    300    FORMAT (I5)
           X=0.
           TBS(1)=0.
           IF (NS.EQ.1) GO TO 3
           DO 5 I=2,NS
           X=X+TBD(I-1)
```

```
5       TBS(I)=X
3       YI=0.
        TI=0.
        VI=0.
        XI=0.
        PSI=PI/2.
        ID=1
        VSR=(VSLR-1)/VSLR
        AL=0.
25      IF (ICVS.EQ.1) ICVS = 2
        IF (ICVS.GE.1) GO TO 210
        IF (IPR.GT.1) GO TO 210
        TYPE 190
190     FORMAT (' ENTER VALUES FOR CA,TLZ,TLD')
        READ (5,200)CA
        READ (5,200) TLZ
        READ (5,200) TLD
        TYPE 225
225     FORMAT (' ENTER VALUES FOR NT,IDP,ICE,IPR,IBS,JCV,IMA,
        1JMS')
        READ (5,500) NT,IDP,ICE,IPR,IBS,JCV,IMA,JMS
500     FORMAT (8I1)
        IF (JMS.EQ.0) GO TO 303
        TYPE 120
        READ (5,100) NS
700     FORMAT('  INT. TIME(SEC.)  SPEED(K/S)  HEADING(R)   ALT.(KM
        1.)  GR.RNG(KM)')
303     IF (IPR.LT.2) GO TO 210
        TYPE 195
195     FORMAT (' ENTER VALUES FOR PSSS,EPS')
        READ (5,200) PSSS
        READ (5,200) EPS
        YO=PSF-PSSS
210     CONTINUE
        DO 10 II=1,NT
        I=ICE+II
        IJK=I
        DDI=NI(I)
        H=TBD(I)/DDI
        VT=TH(I)*4.4482
        S=PI*DI(I)*DI(I)/4.
        NN=NI(I)
        AM=SMZ(I)/2.20462
        III=0
        IF (IPR.GT.1) GO TO 412
        TYPE 700
412     CONTINUE
        DO 20 J=1,NN
        III=J
        IF (VI) 22,21,22
21      T=VT/VSLR
        FD=0.
        PJ1=0.
        YL1=0.
        XM1=0.
        CALL VINCR(PSI,T,AM,YI,FD,AL,VK)
        VK1=H*VK
        GO TO 23
22      CALL DEN(ID,YI)
        CALL FORCES(VI,AL,S,T,FD,FL)
        CALL VINCR(PSI,T,AM,YI,FD,AL,VK)
        CALL PSINCR(VI,PSI,T,AM,YI,FL,AL,PSS)
```

```
      CALL ALT (VI,PSI,YY)
      CALL GRRG(VI,PSI,YI,X)
      VK1=H*VK
      PJ1=H*PSS
      YL1=H*YY
      XM1=H*X
23    TB=TI+H/2.
      V1=VI+VK1/2.
      PS1=PSI+PJ1/2.
      Y1=YI+YL1/2.
      CALL DEN(ID,Y1)
      CALL TILT (CA,TLZ,TLD,TB,AL)
      CALL MASS(TB,I,AM)
      CALL FORCES(V1,AL,S,T,FD,FL)
      CALL VINCR(PS1,T,AM,Y1,FD,AL,VK)
      CALL PSINCR(V1,PS1,T,AM,Y1,FL,AL,PSS)
      CALL ALT (V1,PS1,YY)
      CALL GRRG(V1,PS1,Y1,X)
      VK2=H*VK
      PJ2=H*PSS
      YL2=H*YY
      XM2=H*X
      V2=VI+VK2/2.
      PS2=PSI+PJ2/2.
      Y2=YI+YL2/2.
      CALL DEN(ID,Y2)
      CALL FORCES(V2,AL,S,T,FD,FL)
      CALL VINCR(PS2,T,AM,Y2,FD,AL,VK)
      CALL PSINCR(V2,PS2,T,AM,Y2,FL,AL,PSS)
      CALL ALT (V2,PS2,YY)
      CALL GRRG(V2,PS2,Y2,X)
      VK3=H*VK
      PJ3=H*PSS
      YL3=H*YY
      XM3=H*X
      V3=VI+VK3
      PS3=PSI+PJ3
      Y3=YI+YL3
      TB=TI+H
      CALL DEN(ID,Y3)
      CALL TILT (CA,TLZ,TLD,TB,AL)
      CALL MASS(TB,I,AM)
      CALL FORCES(V3,AL,S,T,FD,FL)
      CALL VINCR(PS3,T,AM,Y3,FD,AL,VK)
      CALL PSINCR(V3,PS3,T,AM,Y3,FL,AL,PSS)
      CALL ALT (V3,PS3,YY)
      CALL GRRG(V3,PS3,Y3,X)
      VK4=H*VK
      PJ4=H*PSS
      YL4=H*YY
      XM4=H*X
      VF=VI+(VK1+2.*(VK2+VK3)+VK4)/6.
      PSF=PSI+(PJ1+2.*(PJ2+PJ3)+PJ4)/6.
      YF=YI+(YL1+2.*(YL2+YL3)+YL4)/6.
      XF=XI+(XM1+2.*(XM2+XM3)+XM4)/6.
      IF (IPR.GT.0) GO TO 501
      TYPE 500,J,TB,VF,PSF,YF,XF
500   FORMAT (1X,I5,5(1X,E10.4))
501   TI=TB
      VI=VF
      PSI=PSF
      YI=YF
```

```
      XI=XF
20    CONTINUE
      IF (IOP.NE.4) GO TO 33
      IF (II.NE.NT) GO TO 10
      ICD=1
      IF (I.EQ.NS) ICD=2
      IF (ICVG.EQ.2) ICD=1
      CALL STORE(VI,PSI,YI,XI,TI,VVI,PSSI,YYI,XXI,TTI,ID,IID,I
CD)
      GO TO 32
33    IF (I.GE.NS) GO TO 29
10    CONTINUE
29    ID=1
32    IC=ICVG
      ICVG=0
      IF (IC.GE.1) GO TO 31
      IF (IPR.EQ.0) GO TO 31
      TYPE 500,III,TB,VF,PSF,YF,XF
      IF (IPR.EQ.1) GO TO 31
      YN=PSF-PSSS
      IF (JCV.EQ.2) GO TO 215
      XN=CA
      GO TO 220
215   XN=TBD(NS)
220   CALL CNVG(XN,XO,YO,YN,EPS,IPR,ICVG)
      TYPE 156,XN,YN
156   FORMAT (' XN EQ. ',E14.8, ' , YO EQ. ',E10.4)
      IF (JCV.EQ.2) GO TO 235
      CA=XN
      GO TO 315
235   TLS=XN
      GO TO 317
31    IF (IC.EQ.2) GO TO 315
      IF (JCV.EQ.2) GO TO 310
      XO=CA
      GO TO 317
310   XO=TBD(IJK)
      TYPE 154
154   FORMAT (' ENTER TLS')
      READ (5,200) TLS
315   IF (JCV.NE.2) GO TO 316
      IF (IJK.EQ.NNS) GO TO 317
      TTT=TLS-TBD(IJK)
      NAB=IJK+1
      DO 316 I=NAB,NNS
316   TBS(I)=TBS(I)+TTT
317   TBD(IJK)=TLS
315   IF (IC.LT.2) GO TO 317
316   IOP=IBS
      IF (IMA.NE.1) GO TO 317
      TYPE 415
415   FORMAT (' ENTER SMIN')
      READ (5,200) SMIN
      DO 417 I=1,NS
417   SMZ(I)=SMZ(I)+SMIN
317   GO TO (30,3,69,25,2) IOP
30    CONTINUE
      END
      SUBROUTINE MASS(TB,I,AM)
      COMMON/C1/SMZ(5),SMD(5),TBS(5),TBD(5),DI(5),TH(5),NI(5)
      SM=SMZ(I)-SMD(I)*(TB-TBS(I))
      AM=SM/2.20462
```

```
      RETURN
      END
      SUBROUTINE TILT (CA,TLZ,TLD,TB,AL)
      AL=0.
      IF (TB.GT.TLZ) GO TO 1
      GO TO 5
1     TT=TLZ+TLD
      IF (TB.GT.TT) GO TO 5
3     T=TB-TLZ
      AL=CA*T*(1.-T/TLD)
5     RETURN
      END
      SUBROUTINE DEN(ID,Y)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      COMMON/B1/YZ(9),YD(9),RHZ(9),ALZ(9),HZ(9),BT(9),PR(9)
      IF (ID.GT.9) GO TO 10
5     IF (Y.LE.YD(ID)) GO TO 3
      ID=ID+1
      IF (ID.LE.9) GO TO 5
10    RHO=0.
      P=0.
      GO TO 7
3     IF (ID.LT.2) GO TO 4
      YY=YD(ID-1)
      IF (Y.GT.YY) GO TO 4
      ID=ID-1
      GO TO 3
4     X=Y-YZ(ID)
      T=BT(ID)
      H=HZ(ID)+T*X
      C=HZ(ID)/H
      A=RHZ(ID)/(1.+ALZ(ID)*X)
      AP=PR(ID)
      B=C**((1.+T)/T)
      BP=C**(1./T)
      RHO=A*B*1000.
      P=AP*BP/10.
      SS=SQRT(1.4*P/RHO)
      SS=SS/1000.
7     RETURN
      END
      SUBROUTINE PSINCR(V,PS,T,AM,Y,FL,AL,PSS)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      X=R+Y
      B=X*X
      PSS=GC/(V*B)-V/X
      PSS=-PSS*COS(PS)
      CD=FL+T
      PSS=PSS+CD*SIN(AL)/(AM*V*1000.)
      RETURN
      END
      SUBROUTINE VINCR(PS,T,AM,Y,FD,AL,VD)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      X=R+Y
      X=X*X
      IF(AL.EQ.0.) GO TO 5
      TT=T*COS(AL)-FD
      GO TO 7
5     TT=T-FD
7     VD=TT/(AM*1000.)-(GC/X)*SIN(PS)
      RETURN
      END
```

```
      SUBROUTINE ALT (V,PS,YD)
      YD=V*SIN(PS)
      RETURN
      END
      SUBROUTINE GRRS(V,PS,Y,X)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      X=V*COS(PS)
      XX=COS(PS)
      X=X*R/(R+Y)
      RETURN
      END
      SUBROUTINE CNVG(XN,XO,YO,YN,EPS,IPR,ICVG)
      DY=(YN-YO)/(XN-XO)
      DX=-YN/DY
      XO=XN
      YO=YN
      XN=XN+DX
      A=ABS(XO-XN)
      IF (A.GE.EPS) GO TO 5
      IPR=0
      ICVG=1
    5 RETURN
      END
      SUBROUTINE STORE(VI,PSI,YI,XI,TI,VVI,PSSI,YYI,XXI,TTI,ID
,IID,ICD
      1)
      GO TO (3,5) ICD
    3 VVI=VI
      PSSI=PSI
      YYI=YI
      XXI=XI
      TTI=TI
      IID=ID
      GO TO 7
    5 VI=VVI
      PSI=PSSI
      YI=YYI
      XI=XXI
      TI=TTI
      ID=IID
    7 RETURN
      END
      SUBROUTINE FORCES(V,AL,S,T,FD,FL)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      T=VT*(1.-VSR*P/PRZ)
      IF (RHO.EQ.0.) GO TO 5
      XM=V/SS
      A=XM-1.14
      IF (XM.GT.0.88889) GO TO 2
      B=-16.6426*A*A
      CD=.25+.3773*EXP(B)
      GO TO 3
    2 IF (XM.GT.1.28) GO TO 4
      B=ABS(A)
      B=B**3.5
      CD=.6273-30.902*B
      GO TO 3
    4 CD=.17913+.5318/XM
    3 IF (XM.GT.1.5) GO TO 6
      CL=4.6-0.5*COS(2.856*XM)
      CDL=1.822-0.298*COS(2.89281*XM)
      GO TO 5
    6 CL=0.8+6./XM
```

```
      CDL=-.2818+3.2727/XM
    5 C=RHO*V*V/2
      C=C*S*1000000.
      FD=CD+CDL*AL*AL
      FD=FD*C
      FL=CL*C
      RETURN
      END
```

APPENDIX C
PDP-11/70 Version of RALLY 6

```
      COMMON/A1/GC,R,SS,VSP,RHO,PRZ,P,VT
      COMMON/B1/YZ(9),YD(9),RHZ(9),ALZ(9),HZ(9),BT(9),PR(9)
      COMMON/C1/SMZ(6),SMD(6),TBS(6),TBD(6),DI(6),TH(6),NI(6)
      DATA YD/10.,25.,45.,55.,85.,90.,100.,120.,140./
      DATA YZ/10.,10.,25.,55.,85.,85.,90.,100.,120./
      DATA RHZ/4.176E-04,4.176E-04,4.048E-05,5.650E-07,9.193E-09,
     19.193E-09,3.842E-09,6.642E-10,3.613E-11/
      DATA ALZ/-3.1308E-04,-3.1308E-04,-3.1235E-04,-3.1089E-04,-3.0945E-04,
     1-3.0945E-04,-3.0921E-04,-3.0873E-04,-3.0778E-04/
      DATA HZ/6.4087,6.4087,6.4388,7.8707,5.7235,5.7235,5.7323,
     16.3487,10.7986/
      DATA BT/-2.0378E-01,2.0064E-03,6.994E-02,2.4469E-03,-7.2257E-02,
     11.7711E-03,6.1446E-02,2.6826E-01,2.3350E-01/
      DATA PR/2.614E05,2.614E05,2.534E04,4.283E02,5.02,5.02,2.098,
     14.005E-01,4.324E-02/
      GC=3.986012E05
      R=6378.145
      PI=3.141592654
      ITY=0
      IPR=0
      ICVG=0
      PRZ=1.013255E05
      TYPE 110
110   FORMAT (' OUTPUT TO LINE PRINTER? (1=YES, 0=NO)')
      READ (5,100) IPR
2     TYPE 120
120   FORMAT (' ENTER NS')
      READ (5,100) NS
100   FORMAT (I1)
      NNS=NS
      TYPE 130,NS
130   FORMAT (' ENTER ',I1,' VALUES FOR SMZ')
      READ (5,200) (SMZ(I),I=1,NS)
200   FORMAT (F10.0)
      TYPE 140,NS
140   FORMAT (' ENTER ',I1,' VALUES FOR SMD')
      READ (5,200) (SMD(I),I=1,NS)
      TYPE 150,NS
150   FORMAT (' ENTER ',I1,' VALUES FOR VAC. TH.')
      READ (5,200) (TH(I),I=1,NS)
      TYPE 152
152   FORMAT (' ENTER VSLP')
      READ (5,200) VSLP
      TYPE 160,NS
160   FORMAT (' ENTER ',I1,' VALUES FOR TBD')
      READ (5,200) (TBD(I),I=1,NS)
      TYPE 170,NS
170   FORMAT (' ENTER ',I1,' VALUES FOR DI')
      READ (5,200) (DI(I),I=1,NS)
69    TYPE 180,NS
180   FORMAT (' ENTER ',I1,' VALUES FOR NI')
      READ (5,300) (NI(I),I=1,NS)
300   FORMAT (I5)
      X=0.
      TBS(1)=0.
      IF (NS.EQ.1) GO TO 3
      DO 5 I=2,NS
      X=X+TBD(I-1)
5     TBS(I)=X
3     YI=0.
      TI=0.
      VI=0.
```

```
      XI=0.
      PSI=PI/2.
      ID=1
      VSR=(VSLR-1)/VSLR
      AL=0.
25    IF (ICVG.EQ.1) ICVG=2
      IF (ICVG.GE.1) GO TO 210
      IF (ITY.GT.1) GO TO 210
      TYPE 190
190   FORMAT (' ENTER VALUES FOR CA,TLZ,TLD')
      READ (5,200) CA
      READ (5,200) TLZ
      READ (5,200) TLD
      TYPE 225
225   FORMAT (' ENTER VALUES FOR NT,IOP,ICE,ITY,IBG,JCV,IMA,JNS')
      READ (5,500) NT,IOP,ICE,ITY,IBG,JCV,IMA,JNS
500   FORMAT (8I1)
      IF (JNS.EQ.0) GO TO 803
      TYPE  120
      PRINT 120
      READ (5,100) NS
700   FORMAT (2X,4HINT.,1X,10HTIME(SEC.),1X,10HSPEED(K/S),1X,
     110HHEADING(R),2X,9HALT.(KM.),1X,10HGR.RNG(KM))
803   IF (ITY.LT.2) GO TO 210
      TYPE 195
195   FORMAT (' ENTER VALUES FOR PSSS,FPS')
      READ (5,200) PSSS
      READ (5,200) FPS
      YO=PSF-PSSS
210   CONTINUE
      DO 10 II=1,NT
      I=ICE+II
      IJK=I
      DDI=NI(I)
      H=TBD(I)/DDI
      VT=TH(I)*4.4482
      S=PI*DI(I)*DI(I)/4.
      NN=NI(I)
      AM=SMZ(I)/2.20462
      III=0
      IF (ITY.GT.1) GO TO 412
      TYPE 700
      IF (IPR.EQ.1) PRINT 700
412   CONTINUE
      DO 20 J=1,NN
      III=J
      IF (VI) 22,21,22
21    T=VT/VSLR
      FD=0.
      PJ1=0.
      YL1=0.
      XM1=0.
      CALL VINCR(PSI,T,AM,VI,FD,AL,VK)
      VK1=H*VK
      GO TO 23
22    CALL DEN (ID,YI)
      CALL FORCES(VI,AL,S,T,FD,FL)
      CALL VINCR(PSI,T,AM,VI,FD,AL,VK)
      CALL PSINCR(VI,PSI,T,AM,YI,FL,AL,PSS)
      CALL ALT (VI,PSI,YY)
      CALL GRRG (VI,PSI,YI,X)
      VK1=H*VK
      PJ1=H*PSS
      YL1=H*YY
      XM1=H*X
23    TR=TI+H/2.
```

```
      V1=VI+VK1/2.
      PS1=PSI+PJ1/2.
      Y1=YI+YL1/2.
      CALL DEN(ID,Y1)
      CALL TILT (CA,TLZ,TLD,TB,AL)
      CALL MASS(TB,I,AM)
      CALL FORCES(V1,AL,S,T,FD,FL)
      CALL VINCR(PS1,T,AM,Y1,FD,AL,VK)
      CALL PSINCR(V1,PS1,T,AM,Y1,FL,AL,PSS)
      CALL ALT (V1,PS1,YY)
      CALL GRPG (V1,PS1,Y1,X)
      VK2=H*VK
      PJ2=H*PSS
      YL2=H*YY
      XM2=H*X
      V2=VI+VK2/2.
      PS2=PSI+PJ2/2.
      Y2=YI+YL2/2.
      CALL DEN(ID,Y2)
      CALL FORCES(V2,AL,S,T,FD,FL)
      CALL VINCR(PS2,T,AM,Y2,FD,AL,VK)
      CALL PSINCR(V2,PS2,T,AM,Y2,FL,AL,PSS)
      CALL ALT (V2,PS2,YY)
      CALL GRPG (V2,PS2,Y2,X)
      VK3=H*VK
      PJ3=H*PSS
      YL3=H*YY
      XM3=H*X
      V3=VI+VK3
      PS3=PSI+PJ3
      Y3=YI+YL3
      TB=TI+H
      CALL DEN(ID,Y3)
      CALL TILT (CA,TLZ,TLD,TB,AL)
      CALL MASS(TB,I,AM)
      CALL FORCES(V3,AL,S,T,FD,FL)
      CALL VINCR(PS3,T,AM,Y3,FD,AL,VK)
      CALL PSINCR(V3,PS3,T,AM,Y3,FL,AL,PSS)
      CALL ALT (V3,PS3,YY)
      CALL GRPG (V3,PS3,Y3,X)
      VK4=H*VK
      PJ4=H*PSS
      YL4=H*YY
      XM4=H*X
      VF=VI+(VK1+2.*(VK2+VK3)+VK4)/6.
      PSF=PSI+(PJ1+2.*(PJ2+PJ3)+PJ4)/6.
      YF=YI+(YL1+2.*(YL2+YL3)+YL4)/6.
      XF=XI+(XM1+2.*(XM2+XM3)+XM4)/6.
      IF (ITY.GT.0) GO TO 501
      TYPE 600,J,TB,VF,PSF,YF,XF
      IF (IPP.EQ.1) PRINT 600,J,TB,VF,PSF,YF,XF
600   FORMAT (1X,I5,5(1X,F10.4))
501   TI=TB
      VI=VF
      PSI=PSF
      YI=YF
      XI=XF
20    CONTINUE
      IF (IOP.NE.4) GO TO 33
      IF (TI.NE.NT) GO TO 10
      ICD=1
      IF (I.EQ.NS) ICD=2
      IF (ICVG.EQ.2) ICD=1
      CALL STORE (VI,PSI,YI,XI,TI,VVI,PSSI,
     1YYI,XXI,TTI,ID,IID,ICD)
      GO TO 32
```

```
33      IF (I.GE.NS) GO TO 29
10      CONTINUE
29      ID=1
32      IC=ICVG
        ICVG=0
        IF (IC.GE.1) GO TO 31
        IF (ITY.EQ.0) GO TO 31
        TYPE 600,III,TB,VF,PSF,YF,XF
        IF (IPR.EQ.1) PRINT 600,III,TB,VF,PSF,YF,XF
        IF (ITY.EQ.1) GO TO 31
        YN=PSF-PSSS
        IF (JCV.EQ.2) GO TO 215
        XN=CA
        GO TO 220
215     XN=TBD(NS)
220     CALL CNVG(XN,XO,YO,YN,EPS,ITY,ICVG)
        TYPE 156,XN,YN
        IF (IPR.EQ.1) PRINT 156,XN,YN
156     FORMAT (' XN EQ. ',E14.8,' , YO EQ. ',E10.4)
        IF (JCV.EQ.2) GO TO 235
        CA=XN
        GO TO 315
235     TLS=XN
        GO TO 817
31      IF (IC.EQ.2) GO TO 815
        IF (JCV.EQ.2) GO TO 310
        XO=CA
        GO TO 317
310     XO=TBD(IJK)
        TYPE 154
154     FORMAT (' ENTER TLS')
        READ (5,200) TLS
815     IF (JCV.NE.2) GO TO 316
        IF (IJK.EQ.NNS) GO TO 817
        TTT=TLS-TBD(IJK)
        NAB=IJK+1
        DO 816 I=NAB,NNS
816     TBS(I)=TBS(I)+TTT
817     TBD(IJK)=TLS
315     IF (IC.LT.2) GO TO 317
316     IOP=IBC
        IF (IMA.NE.1) GO TO 317
        TYPE 415
415     FORMAT (' ENTER SMIN')
        READ (5,200) SMIN
        DO 417 I=1,NS
417     SMZ(I)=SMZ(I)+SMIN
317     GO TO (30,3,69,25,2) IOP
30      CONTINUE
        END
        SUBROUTINE MASS(TB,I,AM)
        COMMON/C1/SMZ(6),SMD(6),TBS(6),TBD(6),DI(6),TH(6),NI(6)
        SM=SMZ(I)-SMD(I)*(TB-TBS(I))
        AM=SM/2.20462
8       RETURN
        END
        SUBROUTINE TILT (CA,TLZ,TLD,TB,AL)
        AL=0.
        IF (TB.GT.TLZ) GO TO 1
        GO TO 5
1       TT=TLZ+TLD
        IF (TB.GT.TT) GO TO 5
3       T=TB-TLZ
        AL=CA*T*(1.-T/TLD)
5       RETURN
        END
```

```
      SUBROUTINE DEN (ID,Y)
      COMMON/A1/GC,R,SS,VSP,RHO,PRZ,P,VT
      COMMON/B1/YZ(9),YD(9),RHZ(9),ALZ(9),HZ(9),BT(9),PR(9)
      IF (ID.GT.9) GO TO 10
5     IF (Y.LE.YD(ID)) GO TO 3
      ID=ID+1
      IF (ID.LE.9) GO TO 5
10    RHO=0.
      P=0.
      GO TO 7
3     IF (ID.LT.2) GO TO 4
      YY=YD(ID-1)
      IF (Y.GT.YY) GO TO 4
      ID=ID-1
      GO TO 3
4     X=Y-YZ(ID)
      T=BT(ID)
      H=HZ(ID)+T*X
      C=HZ(ID)/H
      A=RHZ(ID)/(1.+ALZ(ID)*X)
      AP=PR(ID)
      B=C**((1.+T)/T)
      BP=C**(1./T)
      RHO=A*B*1000.
      P=AP*BP/10.
      SS=SQRT(1.4*P/RHO)
      SS=SS/1000.
7     RETURN
      END
      SUBROUTINE PSINCR (V,PS,T,AM,Y,FL,AL,PSS)
      COMMON/A1/GC,R,SS,VSP,RHO,PRZ,P,VT
      ANG=1.570796327
      X=R+Y
      B=X*X
      PSS=GC/(V*B)-V/X
      PSS=-PSS*SIN(ANG-PS)
      CD=FL+T
      PSS=PSS+CD*SIN(AL)/(AM*V*1000.)
      RETURN
      END
      SUBROUTINE VINCR (PS,T,AM,Y,FD,AL,VD)
      COMMON/A1/GC,R,SS,VSP,RHO,PRZ,P,VT
      ANG=1.570796327
      X=R+Y
      X=X*X
      IF (AL.EQ.0.) GO TO 5
      TT=T*SIN(ANG-AL)-FD
      GO TO 7
5     TT=T-FD
7     VD=TT/(AM*1000.)-(GC/X)*SIN(PS)
      RETURN
      END
      SUBROUTINE ALT (V,PS,YD)
      YD=V*SIN(PS)
      RETURN
      END
      SUBROUTINE GRNG (V,PS,Y,X)
      COMMON/A1/GC,R,SS,VSP,RHO,PRZ,P,VT
      ANG=1.570796327
      X=V*SIN(ANG-PS)
      X=X*R/(R+Y)
      RETURN
      END
      SUBROUTINE CNVG(XN,XO,YO,YN,EPS,ITY,ICVG)
      DY=(YN-YO)/(XN-XO)
      DX=-YN/DY
```

```
      XO=XN
      YO=YN
      XN=XN+DX
      A=ABS(XO-XN)
      IF (A.GE.EPS) GO TO 5
      ITY=0
      ICVG=1
5     RETURN
      END
      SUBROUTINE STORE(VI,PSI,YI,XI,TI,VVI,PSSI,
     1YYI,XXI,TTI,IO,IIO,ICD)
      GO TO (3,5) ICD
3     VVI=VI
      PSSI=PSI
      YYI=YI
      XXI=XI
      TTI=TI
      IIO=IO
      GO TO 7
5     VI=VVI
      PSI=PSSI
      YI=YYI
      XI=XXI
      TI=TTI
      IO=IIO
7     RETURN
      END
      SUBROUTINE FORCES(V,AL,S,T,FD,FL)
      COMMON/A1/GC,R,SS,VSR,RHO,PRZ,P,VT
      ANG=1.570796327
      T=VT*(1.-VSR*P/PRZ)
      IF (RHO.EQ.0.) GO TO 5
      XM=V/SS
      A=XM-1.14
      IF (XM.GT.0.88889) GO TO 2
      B=-16.6426*A*A
      CD=.25+.3773*EXP(B)
      GO TO 3
2     IF (XM.GT.1.28) GO TO 4
      B=ABS(A)
      B=B**3.5
      CD=.6273-30.902*B
      GO TO 3
4     CD=.17913+.5318/XM
3     IF (XM.GT.1.5) GO TO 6
      CL=4.6-0.5*SIN(ANG-2.856*XM)
      CDL=1.822-0.298*SIN(ANG-2.89281*XM)
      GO TO 5
6     CL=0.8+6./XM
      CDL=-.2818+3.2727/XM
5     C=RHO*V*V/2.
      C=C*S*1000000.
      FD=CD+CDL*AL*AL
      FD=FD*C
      FL=CL*C
      RETURN
      END
```

APPENDIX D

```
.EXECUTE RALLY6.FOR
LINK:   LOADING
[LNKXCT RALLY6 EXECUTION]
ENTER NS
4
ENTER 4 VALUES FOR SMZ
400000.
136000.
120000.
10000.
ENTER 4 VALUES FOR SMD
2200.
0.
700.
0.
ENTER 4 VALUES FOR VAC. TH
665000.
0.
220000.
0.
ENTER 4 VALUES FOR VSLR
1.15
1.15
1.
1.
ENTER 4 VALUES FOR TBD
120.
10.
150.
10.
ENTER 4 VALUES FOR DI
3.
3.
3.
3.
ENTER 4 VALUES FOR NI
40
10
10
10
ENTER VALUES FOR CA,TLZ,TLD
-.001
10.
25.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
42012
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   10  .2900E+03  .6822E+01  .1261E+01  .6315E+03  .1637E+03
ENTER VAKUES FOR CA,TLZ,TLD
-.005
10.
25.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
42022
ENTER VALUES FOR PSSS,EPS
0.
.000002
   10  .2900E+03  .7729E+01  .3612E+00  .3379E+03  .6333E+03
XN EQ. -.66051209E-02 , YD EQ.  .3612E+00
```

```
   10  .2900E+03  .3117E+01  .1494E+00  .2029E+03  .7331E+03
XN EQ. -.77368819E-02 , YD EQ.   .1494E+00
   10  .2900E+03  .8360E+01  .3100E-01  .1143E+03  .7824E+03
XN EQ. -.80332980E-02 , YD EQ.   .3100E-01
   10  .2900E+03  .8417E+01  .3159E-02  .9207E+02  .7928E+03
XN EQ. -.80669280E-02 , YD EQ.   .3159E-02
   10  .2900E+03  .8423E+01  .6940E-04  .3958E+02  .7940E+03
XN EQ. -.80676834E-02 , YD EQ.   .6940E-04
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
    1  .3000E+01  .1349E-01  .1571E+01  .2006E-01  .0000E+00
    2  .6000E+01  .2774E-01  .1571E+01  .8172E-01  .0000E+00
    3  .9000E+01  .4279E-01  .1571E+01  .1873E+00  .0000E+00
    4  .1200E+02  .5869E-01  .1566E+01  .3393E+00  .2277E-03
    5  .1500E+02  .7550E-01  .1542E+01  .5403E+00  .3423E-02
    6  .1800E+02  .9328E-01  .1503E+01  .7929E+00  .1546E-01
    7  .2100E+02  .1122E+00  .1454E+01  .1099E+01  .4384E-01
    8  .2400E+02  .1323E+00  .1396E+01  .1462E+01  .9719E-01
    9  .2700E+02  .1540E+00  .1336E+01  .1882E+01  .1846E+00
   10  .3000E+02  .1773E+00  .1275E+01  .2361E+01  .3151E+00
   11  .3300E+02  .2024E+00  .1219E+01  .2900E+01  .4969E+00
   12  .3600E+02  .2294E+00  .1166E+01  .3501E+01  .7368E+00
   13  .3900E+02  .2583E+00  .1115E+01  .4165E+01  .1042E+01
   14  .4200E+02  .2890E+00  .1066E+01  .4892E+01  .1421E+01
   15  .4500E+02  .3209E+00  .1017E+01  .5681E+01  .1883E+01
   16  .4800E+02  .3544E+00  .9699E+00  .6530E+01  .2435E+01
   17  .5100E+02  .3901E+00  .9239E+00  .7435E+01  .3087E+01
   18  .5400E+02  .4285E+00  .8796E+00  .8397E+01  .3847E+01
   19  .5700E+02  .4696E+00  .8370E+00  .9416E+01  .4727E+01
   20  .6000E+02  .5135E+00  .7963E+00  .1049E+02  .5735E+01
   21  .6300E+02  .5605E+00  .7576E+00  .1162E+02  .6881E+01
   22  .6600E+02  .6107E+00  .7208E+00  .1280E+02  .8177E+01
   23  .6900E+02  .6641E+00  .6860E+00  .1404E+02  .9632E+01
   24  .7200E+02  .7208E+00  .6530E+00  .1532E+02  .1126E+02
   25  .7500E+02  .7809E+00  .6220E+00  .1666E+02  .1306E+02
   26  .7800E+02  .8444E+00  .5927E+00  .1805E+02  .1506E+02
   27  .8100E+02  .9115E+00  .5652E+00  .1949E+02  .1726E+02
   28  .8400E+02  .9824E+00  .5393E+00  .2099E+02  .1967E+02
   29  .8700E+02  .1057E+01  .5150E+00  .2252E+02  .2230E+02
   30  .9000E+02  .1135E+01  .4921E+00  .2411E+02  .2517E+02
   31  .9300E+02  .1218E+01  .4706E+00  .2574E+02  .2829E+02
   32  .9600E+02  .1305E+01  .4504E+00  .2742E+02  .3166E+02
   33  .9900E+02  .1396E+01  .4314E+00  .2915E+02  .3531E+02
   34  .1020E+03  .1492E+01  .4136E+00  .3092E+02  .3924E+02
   35  .1050E+03  .1593E+01  .3970E+00  .3274E+02  .4347E+02
   36  .1080E+03  .1698E+01  .3813E+00  .3462E+02  .4801E+02
   37  .1110E+03  .1810E+01  .3666E+00  .3654E+02  .5288E+02
   38  .1140E+03  .1927E+01  .3529E+00  .3851E+02  .5810E+02
   39  .1170E+03  .2050E+01  .3400E+00  .4053E+02  .6367E+02
   40  .1200E+03  .2181E+01  .3280E+00  .4261E+02  .6963E+02
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
    1  .1210E+03  .2177E+01  .3241E+00  .4331E+02  .7168E+02
    2  .1220E+03  .2174E+01  .3202E+00  .4400E+02  .7373E+02
    3  .1230E+03  .2171E+01  .3163E+00  .4468E+02  .7579E+02
    4  .1240E+03  .2168E+01  .3124E+00  .4535E+02  .7783E+02
    5  .1250E+03  .2165E+01  .3085E+00  .4601E+02  .7987E+02
    6  .1260E+03  .2162E+01  .3045E+00  .4666E+02  .8192E+02
    7  .1270E+03  .2159E+01  .3006E+00  .4731E+02  .8397E+02
    8  .1280E+03  .2156E+01  .2966E+00  .4794E+02  .8602E+02
    9  .1290E+03  .2153E+01  .2927E+00  .4857E+02  .8806E+02
   10  .1300E+03  .2150E+01  .2887E+00  .4919E+02  .9011E+02
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
    1  .1450E+03  .2394E+01  .2382E+00  .5732E+02  .1227E+03
```

```
    2  .1600E+03  .2675E+01  .1822E+00  .6569E+02  .1595E+03
    3  .1750E+03  .2997E+01  .1385E+00  .7243E+02  .2010E+03
    4  .1900E+03  .3369E+01  .1010E+00  .7809E+02  .2478E+03
    5  .2050E+03  .3803E+01  .5931E-01  .8261E+02  .3007E+03
    6  .2200E+03  .4316E+01  .4337E-01  .8599E+02  .3605E+03
    7  .2350E+03  .4939E+01  .2306E-01  .8824E+02  .4288E+03
    8  .2500E+03  .5725E+01  .8446E-02  .8944E+02  .5074E+03
    9  .2650E+03  .6787E+01 -.1304E-03  .8977E+02  .5995E+03
   10  .2800E+03  .8424E+01 -.1712E-02  .8959E+02  .7109E+03
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
    1  .2810E+03  .8424E+01 -.1541E-02  .8958E+02  .7192E+03
    2  .2820E+03  .8424E+01 -.1370E-02  .8957E+02  .7275E+03
    3  .2830E+03  .8424E+01 -.1198E-02  .8956E+02  .7359E+03
    4  .2840E+03  .8424E+01 -.1027E-02  .8955E+02  .7442E+03
    5  .2850E+03  .8424E+01 -.8559E-03  .8954E+02  .7525E+03
    6  .2860E+03  .8424E+01 -.6846E-03  .8953E+02  .7609E+03
    7  .2870E+03  .8424E+01 -.5134E-03  .8953E+02  .7691E+03
    8  .2880E+03  .8424E+01 -.3422E-03  .8953E+02  .7774E+03
    9  .2890E+03  .8424E+01 -.1710E-03  .8952E+02  .7857E+03
   10  .2900E+03  .8424E+01  .2314E-06  .8952E+02  .7940E+03
ENTER VALUES FOR CA,TLZ,TLD
-.005
10.
25.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24012
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   10  .1300E+03  .2033E+01  .5209E+00  .6498E+02  .7492E+02
ENTER VALUES FOR CA,TLZ,TLD
-.01
135.
140.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24212
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   10  .2900E+03  .8055E+01 -.9422E-01  .1141E+03  .7199E+03
ENTER VALUES FOR CA,TLZ,TLD
-.008
135.
140.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24222
ENTER VALUES FOR PSSS,EPS
0.
.000002
   10  .2900E+03  .8063E+01 -.3063E-01  .1420E+03  .7199E+03
XN EQ. -.70368524E-02 , YD EQ. -.3063E-01
   10  .2900E+03  .8063E+01 -.7005E-04  .1556E+03  .7195E+03
XN EQ. -.70346443E-02 , YD EQ. -.7005E-04
   10  .2900E+03  .8063E+01 -.4871E-07  .1556E+03  .7195E+03
XN EQ. -.70346427E-02 , YD EQ. -.4871E-07
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
    1  .1450E+03  .2247E+01  .4631E+00  .8012E+02  .1029E+03
    2  .1600E+03  .2496E+01  .3992E+00  .9496E+02  .1347E+03
    3  .1750E+03  .2786E+01  .3317E+00  .1091E+03  .1711E+03
    4  .1900E+03  .3124E+01  .2635E+00  .1220E+03  .2126E+03
    5  .2050E+03  .3524E+01  .1972E+00  .1333E+03  .2601E+03
    6  .2200E+03  .4007E+01  .1355E+00  .1426E+03  .3145E+03
    7  .2350E+03  .4604E+01  .8089E-01  .1495E+03  .3771E+03
    8  .2500E+03  .5373E+01  .3663E-01  .1537E+03  .4499E+03
```

```
   9  .2650E+03  .6427E+01  .7117E-02  .1555E+03  .5359E+03
  10  .2900E+03  .8063E+01 -.7597E-03  .1556E+03  .6407E+03
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   1  .2810E+03  .8063E+01 -.6839E-03  .1556E+03  .6486E+03
   2  .2820E+03  .8063E+01 -.6078E-03  .1556E+03  .6565E+03
   3  .2830E+03  .8063E+01 -.5318E-03  .1556E+03  .6644E+03
   4  .2840E+03  .8063E+01 -.4558E-03  .1556E+03  .6722E+03
   5  .2850E+03  .8063E+01 -.3799E-03  .1556E+03  .6801E+03
   6  .2860E+03  .8063E+01 -.3039E-03  .1556E+03  .6880E+03
   7  .2870E+03  .8063E+01 -.2279E-03  .1556E+03  .6958E+03
   8  .2880E+03  .8063E+01 -.1519E-03  .1556E+03  .7037E+03
   9  .2890E+03  .8063E+01 -.7597E-04  .1556E+03  .7116E+03
  10  .2900E+03  .8063E+01  .2295E-08  .1556E+03  .7195E+03
ENTER VALUES FOR CA,TLZ,TLD
-.005
10.
25.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24012
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
  10  .1300E+03  .1973E+01  .6559E+00  .7227E+02  .6555E+02
ENTER VALUES FOR CA,TLZ,TLD
-.01
135.
140.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24212
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
  10  .2900E+03  .7809E+01  .2714E-01  .2000E+03  .6698E+03
ENTER VALUES FOR CA,TLZ,TLD
-.013
135.
140.
ENTER VALUES FOR NT,IOP,ICE,IPR,IBG,JCV,IMA,JNS
24222
ENTER VALUES FOR PSSS,EPS
0.
.000002
  10  .2900E+03  .7775E+01 -.7398E-01  .1574E+03  .6690E+03
XN EQ. -.10805297E-01 , YD EQ. -.7398E-01
  10  .2900E+03  .7801E+01  .5980E-04  .1886E+03  .6701E+03
XN EQ. -.10807070E-01 , YD EQ.  .5980E-04
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   1  .1450E+03  .2170E+01  .5991E+00  .9050E+02  .9034E+02
   2  .1600E+03  .2400E+01  .5296E+00  .1088E+03  .1188E+03
   3  .1750E+03  .2667E+01  .4481E+00  .1266E+03  .1517E+03
   4  .1900E+03  .2979E+01  .3624E+00  .1432E+03  .1898E+03
   5  .2050E+03  .3350E+01  .2757E+00  .1581E+03  .2337E+03
   6  .2200E+03  .3803E+01  .1924E+00  .1704E+03  .2845E+03
   7  .2350E+03  .4373E+01  .1169E+00  .1797E+03  .3433E+03
   8  .2500E+03  .5123E+01  .5437E-01  .1856E+03  .4120E+03
   9  .2650E+03  .6167E+01  .1189E-01  .1882E+03  .4937E+03
  10  .2900E+03  .7801E+01 -.3136E-04  .1885E+03  .5944E+03
 INT. TIME(SEC.) SPEED(K/S) HEADING(R)  ALT.(KM.) GR.RNG(KM)
   1  .2810E+03  .7801E+01 -.2821E-04  .1885E+03  .6019E+03
   2  .2820E+03  .7801E+01 -.2506E-04  .1885E+03  .6095E+03
   3  .2830E+03  .7801E+01 -.2191E-04  .1885E+03  .6171E+03
   4  .2840E+03  .7801E+01 -.1876E-04  .1885E+03  .6247E+03
   5  .2850E+03  .7801E+01 -.1561E-04  .1885E+03  .6323E+03
   6  .2860E+03  .7801E+01 -.1246E-04  .1885E+03  .6399E+03
```

```
    7  .2870E+03  .7801E+01 -.9312E-05  .1885E+03  .6474E+03
    8  .2880E+03  .7801E+01 -.6162E-05  .1885E+03  .6550E+03
    9  .2890E+03  .7801E+01 -.3013E-05  .1885E+03  .6626E+03
   10  .2900E+03  .7801E+01  .1361E-06  .1885E+03  .6701E+03
ENTER VALUES FOR CA,TLZ,TLD
^C

.K/F
```

DEPARTMENT OF THE NAVY

NAVAL RESEARCH LABORATORY
Washington, D.C. 20375

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

POSTAGE AND FEES PAID
DEPARTMENT OF THE NAVY
DoD-316

THIRD CLASS MAIL

U.S.MAIL

NRL

THE NAVAL RESEARCH LABORATORY IS AN ACTIVITY OF THE OFFICE OF NAVAL RESEARCH